Congestion Removal in the Next Generation Internet

Robert Suryasaputra

Doctor of Philosophy

2007

RMIT

Congestion Removal in the Next Generation Internet

Robert Suryasaputra

B. Eng. (Hons)

A thesis submitted in fulfilment of the requirements for

the degree of Doctor of Philosophy

School of Electrical and Computer Engineering

Science, Engineering and Technology Portfolio

RMIT University

Melbourne, Victoria, Australia

March 2007

Declaration

I certify that except where due acknowledgement has been made, the work is that of the author alone; the work has not been submitted previously, in whole or in part, to qualify for any other academic award; the content of the thesis is the result of work which has been carried out since the official commencement date of the approved research program; and, any editorial work, paid or unpaid, carried out by a third party is acknowledged.

Robert Suryasaputra

Melbourne, August 20, 2007

Congestion Removal in the Next Generation Internet

To my family

Acknowledgement

Beginning in early 2003 until the present time has been a long journey with many ups and downs. Thankfully, with uncountable help and support from many people, I have managed to survive so far. Therefore, I would like to express my deepest honour and appreciation to these wonderful people.

First, my deepest thanks go to Prof. Richard J. Harris, currently Chair of Telecommunications and Network Engineering at Massey University, NZ for supervising me. Without his guidance and advice, I would not be writing this PhD thesis, but rather a submission for Masters by Research. I am very grateful for being one of his students. Having a mentor like Richard, from whom I have learnt a great deal, is something that I dream of as being of benefit for all RMIT research students.

Other people that have been supporting me throughout this candidature are Dr. John Murphy, Dr. Alexander Kist and Dr. Bill Lloyd Smith who have all been part of the RMIT CATT Centre during this time. John guided me with regard to English grammar and writing skills from the start. Through long discussions and many paper revisions, John included me as one of his co-authors in two early conference papers, which provided me with quick publications as a first year PhD research student. He also contributed greatly to my way of problem solving. Bill has been a reliable source of information that answered my questions about the fundamentals of Linear Programming and

Statistics. Alex has provided reviews and advice – especially concerning the significance of published papers and how they should be presented.

The CATT Centre family were always a great source of inspiration and invaluable help. In the early days of my candidature, Andrew Cassin was there to help and explain to me about what was going on. He has been my computer "guru". Hieu Tran, who was a PhD student and member of the CATT Centre family, also has been a great help in programming and networking. I envy him because of his skills and his knowledge. The two senior researchers, Dr. Irena Atov and Dr. Deddy Chandra, also helped me throughout the candidature with useful discussion sessions and helpful advice.

I also would like to thank Suyong Eum, Dr. Suresh Venkatachalaiah, Rebecca Wang, Dr. Kevin Lin and Dr. Jimmy Lau for their friendship and for making my time so much more enjoyable. Being a part of CATT Centre family has also taught me that hard work and dedication is the *only* way to achieve your dreams.

I also thank my parents and my sister for providing me with such a wonderful environment when I was growing up, especially my mum, who has educated me in such a way that I am growing to be what I am now. My sister, who has always been eager to know my story and at the same time appreciates my decisions. To Our Lady of Victories congregation, choir members, Ong's family and friends, I would like to express my thanks for making me part of their big family. The choir has been my delight in serving and praising the Lord. My final thanks go to all my friends during high school and university, who have contributed so much to my life.

Without these wonderful people, I am sure that you would be doing something else rather than reading this thesis, because this thesis would never exist without the continuous support of all these people.

Contents

Abstract

The ongoing development of new and demanding Internet applications requires the Internet to deliver better service levels which are significantly better than the best effort service that the Internet currently provides and was built for. These improved service levels include guaranteed delays, jitter and bandwidth. Through extensive research into Quality of Service and Differentiated Service (DiffServ) it has now become possible to provide guaranteed services; however, this turns out to be inadequate without the application of Traffic Engineering methodologies and principles. Traffic Engineering is an integral part of network operation. Its major goal is to deliver the best performance from an existing service providers' network resources and, at the same time, to enhance a customers' view of network performance.

In this thesis, several different traffic engineering methods for optimising the operation of native IP and IP networks employing MPLS are proposed. A feature of these new methods is their fast run times and this opens the way to making them suitable for application in an online traffic engineering environment. For native IP networks running shortest path based routing protocols, we show that an LP-based optimisation based on the well-known multi-commodity flow problem can be effective in removing network congestion. Having realised that Internet service providers are now migrating their networks to the use of MPLS, we have also formulated optimisation methods

to traffic engineer MPLS networks by selecting suitable routing paths and utilising the feature of explicit routing contained in MPLS.

Although MPLS is capable of delivering traffic engineering across different classes of traffic, network operators still prefer to rely on the proven and simple IP based routing protocols for best effort traffic and only use MPLS to route traffic requiring special forwarding treatment. Based on this fact, we propose a method that optimises the routing patterns applicable to different classes of traffic based on their bandwidth requirements.

A traffic engineering comparison study that evaluates the performance of a neural network-based method for MPLS networks and LP-based weight setting approach for shortest path based networks has been performed using a well-known open source network simulator, called ns2. The comparative evaluation is based upon the packet loss probability. The final chapter of the thesis describes the software development of a network management application called OptiFlow which integrates techniques described in earlier chapters including the LP-based weight setting optimisation methodology; it also uses traffic matrix estimation techniques that are required as input to the weight setting models that have been devised. The motivation for developing OptiFlow was to provide a prototype set of tools that meet the congestion management needs of networking industries (ISPs and telecommunications companies – telcos).

Chapter 1

Introduction

Nowadays, the Internet is used to transport many different kinds of traffic from many different applications. Each of these applications require a specific treatment for their packets. However, the underlying principle, the principle upon which the Internet was built, does not include sufficient provision for special packet treatment. The "best effort" service provided by native IP networks might have been sufficient for delay insensitive traffic such as email or web traffic but the requirements of new types of traffic, that need performance guarantees, require service providers to provision and manage their networks effectively and efficiently [1] [2].

Internet traffic engineering methods can provide a solution for service providers wishing to effectively manage these new traffic types. Internet "traffic engineering" is defined as that aspect of Internet network engineering that deals with the issue of performance evaluation and optimisation of operational IP networks. This definition includes the application of technology and scientific principles to the measurement, characterisation, modelling and control of Internet traffic [3]. The performance improvement wrought by traffic engineering can be at the traffic and resource levels. This is done by ad-

dressing traffic requirements, while utilising network resources economically and reliably.

The optimisation aspects of traffic engineering can be achieved through capacity management and traffic management. Capacity management includes capacity planning, routing control and resource management. Network resources of particular interest include link bandwidth, buffer space and computational resources. Traffic management includes traffic control functions such as traffic conditioning, queue management, scheduling and access control [4].

The Internet conveys information from source nodes to destination nodes in IP packets, this means that the three critical components that allow the Internet to function properly are: addressing, store and forward operation and routing. Accordingly, routing is one of the most significant functions performed by the Internet. Routing involves finding an appropriate path between ingress nodes and egress nodes. Therefore, one of the most important functions performed by Internet traffic engineering is the control and optimisation of the routing function; to direct traffic through the network in the most effective way. Modern intra-domain routing protocols such as OSPF (Open Shortest Path First) or IS-IS (Intermediate Systems-Intermediate Systems) employ shortest route algorithms to define pathways for application packets. The shortest path computations are based on path length data ("metrics") and employ a number of well-known shortest path algorithms due to Dijkstra and Bellman-Ford.

Given the importance of Internet traffic engineering, service providers require tools that allow them to manage the performance of their IP networks. In [5], Feldmann et al envisaged that the current networking industry is lacking in systems that a large service provider can use to support the necessary

traffic measurement and modelling activity. Hence, they presented a network management system, called *NetScope*, which allows network managers to visualise, monitor and influence routing decisions for an operational network. Included in the NetScope system is a network optimisation module that optimises intra-domain routing by manipulating a set of shortest path routing metrics [6].

Before such a formal method was proposed, network operators could influence routing decisions by manually adjusting the set of link weights. It should be noted that this process is error-prone and is also tedious in the case of networks carrying hundreds or even thousands of flows [7]. Furthermore, the network operator can only predict, by intuition, the effect caused by such changes, and this intuition may not always be correct in large and complex networks. This further encourages the need for a formal tool or a system to monitor and manage IP networks and that takes a more *global view* of network requirements.

The major challenge in finding an optimal set of shortest path routing metrics is the computational burden. The problem is difficult and is proven to be an NP-type problem. Hence, heuristics are needed. A Tabu search based search proposed by Fortz and Thorup [6] is one of the heuristics that has been proposed to solve the problem. Other heuristics exploit genetic algorithms and simulated annealing to perform the optimisation. The problem can also be modelled mathematically; however, it clearly becomes unsolvable as the network grows larger. Nonetheless, these approaches do not allow online optimisation and implementation due to the long algorithm run-times involved.

The introduction of Multiprotocol Label Switching [8] allowed more flexible traffic engineering for service providers [9]. The evolvement of new Internet applications that require different and preferential treatment also pushes

the necessity for MPLS deployment in service provider networks. With the trend of network providers upgrading their networks to MPLS networking, it is becoming apparent that future traffic engineering will utilise MPLS or its extensions.

1.1 Focus of this Thesis

In this thesis, we present three different models for network optimisation that take into account some of the technology enhancements used in IP based core networks. The proposed methods allow a rapid deployment of traffic engineering in traditional IP networks or MPLS based networks. Assuming that the network topology is given, the objective of traffic engineering – through a routing strategy – is to determine the optimal routes for IP traffic.

We focus on the problem of routing optimisation in IP networks. Specifically, we are looking at problems of routing inside an autonomous system (AS). An AS is defined as a group of routers exchanging routing information via a common routing protocol. Inside its own AS, a service provider has total control of the routing policies that are used. Within this AS, a service provider typically runs OSPF (Open Shortest Path First) or IS-IS (Intermediate Systems-Intermediate Systems) as the intra-domain routing protocol.

Our first model is applicable to networks running OSPF and IS-IS, which operate on a shortest path paradigm. A router routes packets to their destinations on a path that has the lowest path cost / routing metric / weight (in this thesis, these terms will be used interchangeably). The path metric is a sum of metrics on links that constitute that path. We use an LP-based approach to find out an appropriate set of link weights according to a given demand and network topology. It is shown in [10] [11] [12] that there is an

important relationship between optimal dual variables of the classical capacitated flow allocation problem and the link weights for shortest path routing. However, since the uniqueness of the shortest path cannot be guaranteed, the solution of the weight setting problem might violate the constraints of the routing problem. In this work, we are quantifying, by using a "splitting ratio concept", how much the constraints are violated due to a "non-unique" weight set.

Furthermore, a running (operational) network that requires route optimisation should not have to undergo a large number of configuration changes (i.e. weight changes). The effect of many weight changes might create a significant disturbance to an operational network. Thus, an important question to be answered is: *How to determine an optimised weight set for a running network such that the link load is below a specific threshold?* To investigate the network impact of multiple weight changes, we also simulate these weight changes using the well-known open source network simulator known as **ns-2**.

In our second model, we look at the problem of path selection in an MPLS network using different objectives and constraints. In [13], Gavish et al described the joint problem of selecting a route and a capacity value for each OD-Pair in the network. However, this work was intended for general computer communication networks. The arbitrary path selection is achieved by using explicit routing in MPLS [14] [15]. To some extent, the problem is a variant of the single-source unsplittable flow problem, which is known to be an NP-hard problem [16]. In this thesis, we shall use a mathematical model with two different objective functions and compare their performance in terms of packet loss using the ns-2 simulator. Furthermore, it is also possible to minimise the number of explicitly configured MPLS paths during the optimisation of a running network.

A third model is introduced with the objective of handling multiple classes of traffic from different applications. In order to fulfill the delay and / or jitter requirement of these traffic streams, networks have to be able to allocate and reserve bandwidth for different traffic types as a first step. Traffic streams with different delay and grade of service requirements can be accommodated using an equivalent bandwidth model based on the work described in [17]; although this model is only of limited use for network planning [18], it is the first step towards determining bandwidth requirements for traffic engineering purposes. In our model, different traffic streams have a predefined bandwidth allocation. They are then routed in the network; the higher priority traffic is routed using MPLS explicit routes, while the best effort traffic is routed using underlying IGP mechanisms for simplicity.

To see how much value the new technology, MPLS, has brought to the networking industry, we investigate the performance of two methods for network optimisation that are targeted at different technologies. The interest is to see how much improvement MPLS has contributed in comparison to simply using IP native routing. The performance study is once again performed using the ns-2 simulator. The packet loss probabilities are used as the performance indicators. The simulation is done using a large number of different traffic matrices to simulate traffic demand variations in real networks.

The main challenge in managing IP networks involves an understanding of current traffic flows, routing and network configuration. IP network management includes IP router configuration, traffic monitoring and control. With the under-development of tools in the IP network management area, traffic engineering processes have not been done effectively [5]. We propose a software architecture and implementation of a network management system, called OptiFlow. OptiFlow is capable of monitoring the load levels on all links

in the managed AS network and intervenes by changing the routing when congestion is detected in this network.

1.2 Contributions of this Thesis

In this thesis, methods for optimising IP networks both on-line and off-line are presented. The optimisation models take into account the many different features of networks that can be used to carry out traffic engineering. The contributions of this thesis led to the following publications: [19] [20] [21] [22] [23] [24] [25] [26]. The key contributions of this thesis are summarised below:

The work in Chapter 3 led to the publication of [20] [21]. The contributions of this chapter can be summarised as follows:

- Development of a dual-simplex method for obtaining initial solutions to the multi-commodity flow problem, particularly the determination of a set of initial weights.
- Investigation of the splitting ratios that determine the degree of uniqueness of the solution.
- Study of network routing speed of convergence following weight changes.

The work in Chapter 4 led to the publication of [23] [19]. The contributions of this chapter can be summarised as follows:

- Development of two mathematical models that allow explicit route configuration in MPLS networks.
- ns-2 simulation to compare the packet loss performance of the different models.

- Minimal configuration changes for the re-optimisation of MPLS networks.

The work in Chapter 5 led to the publication of [26]. The contributions of this chapter can be summarised as follows:

- Development of a model and a numerical study of an MIP/LP-based optimisation model that takes into account multiple traffic streams in IP networks. The traffic streams, depending on QoS or bandwidth requirements, are routed using different technologies, i.e. native IP routing or MPLS.

The work in Chapter 6 led to the publication of [24]. The contributions of this chapter can be summarised as follows:

- Description of two methods used in optimising OSPF and MPLS networks. The two methods are: LP-based weight setting and Neural Networks/Marginal Increase Heuristic.

- Comparison study of a medium sized network with a large number of OD-Pairs using ns-2 simulation to investigate the packet loss probabilities.

The work in Chapter 7 led to the publication of [25]. The contributions of this chapter can be summarised as follows:

- Design and implementation of network management software, OptiFlow, a Microsoft Windows application that implements network management functionalities, including traffic matrix estimation and network congestion removal.

- Design and implementation of a virtual network test bed using VMWare, Zebra and Net-SNMP to replace the dependency on a set of physical routers.

Publications by the author

1. J. Murphy, R. Suryasaputra, and R. Harris, "Link Congestion Avoidance Using Weight Setting in an MPLS Network - a Simple LP Approach," in *ATNAC 2003*, Melbourne, Australia, December 2003.

2. B. Lloyd-Smith, R. Suryasputra, J. Murphy, and R. Harris, "Removing hot spots using a simple LP method and why it works," in *ATNAC*, Melbourne, 2003.

3. J. Murphy, R. Suryasaputra, W. Lloyd-Smith, X. V. Doan, R. Nelson, and R. Harris, "Dynamic Resource Management Using LP Weight Setting in an MPLS Domain," in *Next Generation Teletraffic and Wired/Wireless Advanced Networking (NEW2AN'04)*, 2004.

4. R. Suryasaputra, J. Murphy, and R. Harris, "Dousing Hot Spots in OSPF Networks," in *ATNAC 2004*, Sydney, Australia, December 2004.

5. R. Suryasaputra, A. A. Kist, and R. J. Harris, "Verification of MPLS Traffic Engineering Techniques," in *ICON2005. International Conference on Networks*, 2005.

6. R. Suryasaputra, A. A. Kist, and R. J. Harris, "OptiFlow - A Capacity Management Tool," in *TENCON 2005. 2005 IEEE Region 10 Conference. IEEE*, 2005.

7. R. Suryasaputra, A. A. Kist, H. L. Ferra, R. A. Palmer, M. J. Dale, and R. J. Harris, "Comparison of Intra-Domain Traffic Engineering Methods," in *World Teletraffic Congress WTC*, 2006.

8. R. Suryasaputra, A. A. Kist, and R. J. Harris, "Reconfiguration in Multi Class of Services Networks," in *ATNAC*, Melbourne, 2006.

9. S. Eum, R. J. Harris, B. Lloyd-Smith, and R. Suryasaputra, "Dynamic Weight Changes for Traffic Matrix Estimation," in *Networks 2006. Telecommunications Network Strategy and Planning Symposium, 2006. 12th International Conference on Networks*, 2006.

10. S. Venkatachalaiah, R. Suryasaputra, and R. J. Harris, "Improvement of Handoff Performance in Wireless Networks Using Mobility Prediction and Multicasting," in *ICON2005. International Conference on Networks*, 2005.

11. S. Venkatachalaiah, R. J. Harris, and R. Suryasaputra, "Improvement of Handoff Performance in Wireless Networks using Mobility Prediction and Multicasting Techniques," in *in Proc. of 4th WSEAS Int. Conf. on Electronics, Hardware, Wireless & Optical Communications (EHAC 2005)*.

12. S. Venkatachalaiah, R. J. Harris, and R. Suryasaputra, "Improvement of handoff performance in wireless networks using mobility prediction and multicasting techniques," vol. 4, pp. 104–111, 2005.

Throughout the PhD candidature, the author contributed to the publication of the papers listed as 9, 10, 11 and 12 in the above summary, but these topics are not discussed in this thesis.

1.3 Organisation

Chapter 2: To provide the context for network traffic engineering, we start this chapter by first reviewing existing techniques, technologies and mechanisms that are being deployed by network providers in their IP-based net-

works. Firstly, this chapter gives an overview of solutions for native IP networks that work based on the shortest path routing paradigm. MPLS has been introduced as a new technology for IP networks people are now utilising its traffic engineering features to help with the delivery of Quality of Service. Traffic engineering has become more complicated due to the growth of IP networks as transport networks that carry many different types of traffic from many different applications. Herein, we also outline the existing approaches to handling traffic engineering with multiple classes of traffic.

Chapter 3: In this chapter, we use the classical multi-commodity flow problem in order to find a set of shortest path weights. Although this set of weights might not generate unique shortest paths, we show through experimental study that initial solutions play an important role in the overall solution quality. In this chapter, we use the concept of a splitting ratio to quantify the degree of uniqueness of the solution.

Chapter 4: MPLS, Multiprotocol Label Switching, was initially proposed to speed up IP routing. The MPLS traffic engineering feature is described in detail in this chapter. For our techniques, we exploit its explicit routing capability. This allows us to construct an arbitrary set of paths upon which to route traffic. Our numerical study indicates that the model restricts the link utilisation in the network to a certain threshold, thus preventing network congestion. Furthermore, through our simulation study, we show that having implemented the model it reduces packet loss significantly in comparison to native IP routing. We also show that it is possible to reduce the number of explicit routed LSPs during the optimisation process by slightly increasing the maximum link utilisation target in the network.

Chapter 5: In this chapter, we propose a model that optimises a hybrid OSPF/IS-IS and MPLS based network. In the model, high priority traffic is

carried inside an LSP, whilst best effort traffic is routed based on shortest path routing. There are two distinct problems here: The first is to find a set of paths for both types of traffic. The second is to find a set of shortest path weights based on the routing pattern found from the first problem. Both problems are modelled mathematically and solved using LP solvers.

Chapter 6: This chapter presents a comparative study of two different traffic engineering methods that are deployed in two different networks. The first uses LP-based weight setting for native IP networks and the second is an heuristic for MPLS path selection. These two methods are evaluated by using the ns-2 simulator. The topology tested is chosen to be a medium sized network with a large number of traffic streams. The performance benchmark is based on packet loss probabilities.

Chapter 7: This chapter addresses the network management limitations faced, on a daily basis, by network administrators. Having analysed what the networking industry is lacking at the moment, we designed OptiFlow, a network management software tool. OptiFlow is capable of monitoring a running network based on SNMP queries. Furthermore, it enables network administrators to gather information about traffic flows in the network (such as traffic matrix information per origin and destination pair). Having detected congestion in the network, OptiFlow is able to invoke an optimisation module that uses LP-based weight setting to ease the congestion in the network.

Chapter 8: This chapter is devoted to providing some final conclusions and discussing future research directions.

Chapter 2

Background

This chapter addresses problems associated with congestion caused by an uneven traffic distribution in IP networks using traffic engineering solution methods. To address the problem of congestion in the Internet world, it is important to understand the operation of the Internet. A description of how traffic is routed in the Internet is presented in this chapter. Included in this chapter there are descriptions of several techniques that have been proposed to resolve congestion in Internet networks.

In the early days of the Internet, capacity provisioning was an acceptable solution to the problems of congestion in the Internet because, at this time, traffic was generally delay-insensitive (ie. best-effort traffic). However, as the Internet is now used as a transport network for more and more applications that require specific service guarantees, this simplistic approach to capacity provisioning simply does not work anymore. Stricter controls are required to deliver services with a quality that is better than that of best-effort traffic. MPLS is capable of delivering Quality of Service for those requiring special service guarantees; however, MPLS alone is not capable of delivering QoS.

Descriptions of traffic engineering methods that can be employed in na-

tive IP networks (those running OSPF or IS-IS), newer switching technology (MPLS) and hybrid networks (networks operating native IP routing and MPLS simultaneously) are presented in this chapter.

2.1 Routing in the Internet

Routing is loosely defined as a process to get an IP packet from a source to a destination node in the network. However, as currently implemented, it involves three distinct processes, namely: network topology discovery, route computation and packet forwarding. Topology discovery and route computation are done in the router's control plane with the aid of routing protocols, while packet forwarding is performed in a router's forwarding plane by the IP layer. Routers in a particular region collectively run the same routing protocol, which ensures that there is an up-to-date and synchronised network-wide view available. Having obtained a picture of the network, connectivity can be determined and routes can be computed.

Routing protocols that carry out routing functionality in the Internet can be classified according to the scope of the routing performed. Routing protocols that operate inside an autonomous system, wherein routers are under the same administration, are called intra-domain (interior) routing protocols, also referred to collectively as the Interior Gateway Protocol - IGP. On the other hand, exterior gateway protocols provide routing among autonomous systems. Figure 2.1 gives further classification based on how the protocols work together with some examples [27].

Once the routing protocols provide the necessary information, routers can then compute reachability information for IP prefixes (a block of IP addresses). The reachability information may include information such as which one of

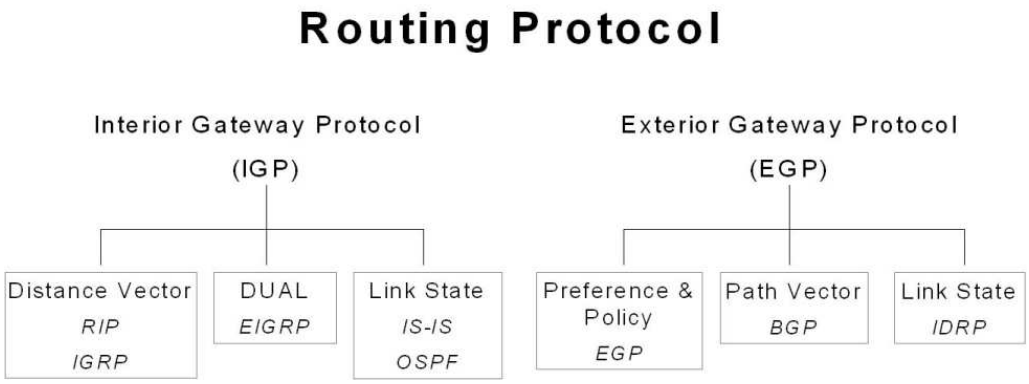

Figure 2.1: Internet Routing Protocols Classification

the neighbouring routers can a router forward a packet to, such that the packet is closer to its destination.

2.2 Traffic Engineering

Traffic Engineering (TE) is a term that describes the process of delivering performance optimisation for operational networks. It covers the application of technology and knowledge in measurement, modelling and control of Internet traffic to achieve specific network performance objectives. A major goal of Internet Traffic Engineering is to enable efficient and reliable network operations while simultaneously optimising network resource utilisation and traffic performance. Traffic Engineering functionality has become a crucial part of large Autonomous Systems because of the high cost of network assets (capital expenditure - CAPEX).

The performance objectives of traffic engineering can be classified as traffic oriented or resource oriented. Traffic oriented performance objectives aim to enhance the Quality of Service of traffic streams. Minimising packet loss is an example of a traffic oriented performance objective in a single class best ef-

fort model, whereas delay variation is another example of such a performance objective in the case of a multi-class traffic model. Resource oriented performance objectives aim to manage bandwidth resources efficiently and ensure balanced loading in a network. Regardless of the classification, both have congestion minimisation as their primary objective.

Network congestion manifests itself in two different situations: firstly, when network resources are insufficient to accommodate the offered load and, secondly, there is inefficient mapping of traffic streams to available network resources. The first situation is addressed by applying classical congestion control techniques or capacity expansion. Examples of congestion control techniques include rate limiting, window flow control, queue management and schedule-based control. The second type of congestion, namely that due to inefficient resource allocation, is addressed by Traffic Engineering.

In the context of Internet Traffic Engineering, the congestion problem is caused by inappropriate or inefficient mapping of available network resources to traffic streams; thus causing some parts of the network resources to become over-utilised while others remain under utilised. Network congestion situations addressed by traffic engineering represent prolonged congestion rather than transient congestion due to instantaneous bursts of traffic.

Generally, congestion due to inefficient resource allocation can be reduced by carrying out load balancing policies. The objective of such policies is to minimise the maximum resource utilisation, thus forcing the load to be spread as evenly as possible. When congestion is minimised, packet losses decrease, delays decrease and aggregate throughput increases. Thereby, the network service quality perceived by end users becomes significantly enhanced.

Performance optimisation of operational networks is an example of a control problem. In this model, a traffic engineering module acts as the controller

of the interconnected network elements such as links, routers and performance monitoring tools. The traffic engineering module observes the current state of the network using the monitoring tools and uses network policies to make decisions on how to bring the network into a desired operating state. The control actions can be carried out through modification of traffic management parameters, routing and constraints associated with these resources.

2.2.1 Intra-domain Traffic Engineering in native IP networks

This subsection describes some of the well-known limitations of current Intra-domain routing protocols with regard to Traffic Engineering principles.

The control capabilities offered by existing Internet interior gateway protocols, such as Open Shortest Path First (OSPF) and Intermediate System-Intermediate System (IS-IS), are not adequate to enable Traffic Engineering. This makes it difficult to perform effective policies to address network congestion problems. Indeed, IGPs based on shortest path first (SPF) algorithms contribute significantly to congestion problems in Autonomous Systems within the Internet. SPF algorithms generally optimise the routing based on a simple additive metric that is set by network administrators. These protocols are topology driven, so bandwidth availability and traffic characteristics are not involved in routing decisions. Consequently, congestion frequently occurs when:

- The shortest paths of multiple traffic streams converge on specific links or router interfaces, or
- A single traffic stream is routed through a link or router interface which does not have enough bandwidth to accommodate it.

These congestions can still occur even when feasible alternate paths with excess capacity exist. It is this aspect of congestion problems (an indicator of suboptimal resource allocation) that Traffic Engineering aims to vigorously resolve. Equal cost path load sharing can be used to address the second cause for congestion listed above with some degree of success, however it is generally not helpful in alleviating congestion due to the first cause. Furthermore, equal cost load sharing is not particularly effective in large networks with a dense topology.

A popular approach to circumvent the inadequacies of current IGPs is to introduce an additional logical layer which is commonly known as an *overlay model*, such as IP over ATM or IP over Frame Relay (FR). The overlay model removes network topological restrictions by enabling arbitrary virtual topologies to be provisioned atop the network's physical topology. The virtual topology is constructed from virtual circuits (VCs) or logical IP links which appear as physical links to the IGP routing protocols. The overlay model provides additional important services to support traffic and resource control, including: (1) constraint-based routing at the VC level, (2) support for administratively configurable explicit VC paths, (3) path compression, (4) call admission control functions, (5) traffic shaping and traffic policing functions, and (6) survivability of VCs. These capabilities enable the actualisation of a variety of Traffic Engineering policies. For example, virtual circuits can easily be rerouted to move traffic from over-utilised resources onto relatively under-utilised ones.

However, the overlay model is complicated. It requires two different management teams; one for the IP layer and the other for the ATM or FR layer. When failures occur, both of the layers and their interactions have to be checked and verified. The resource mapping between the IP and ATM/FR layer needs to be modified when the traffic streams change.

Assuming *a priori* knowledge of traffic demand, it is possible to find a routing pattern in the given network topology such that the performance objectives are satisfied. The routing constraint here is the shortest path constraint, wherein the path to the destination must be a shortest path based on some sort of metric values, which are set by the network administrator. This technique is known as *weight setting*, which involves allocating an appropriate routing metric such that the desired routing pattern is achieved.

2.2.2 MPLS Switching to deliver Traffic Engineering

Multiprotocol Label Switching (MPLS) is strategically significant for Traffic Engineering because it can potentially provide most of the functionality available from the overlay model, in an integrated manner, and at a lower cost than the currently competing alternatives. MPLS also offers the possibility of automating aspects of the Traffic Engineering function. MPLS is a switching technology, which replaces traditional IP packet forwarding by a form of circuit switching. In MPLS switching, an IP packet is encapsulated in an MPLS packet. The decisions involved in passing the IP packet along to the next hop are not based on an IP packet's destination address, but on MPLS labels, which are added to the MPLS packet header.

MPLS also allows for greater routing flexibility. Native IP routing protocols work based on next-hop destination-based routing. When forwarding a packet, a router determines the outgoing interface based solely on the destination address of the packet and forwards the packet to the respective neighbouring router. The neighbouring router handles the packet in the same way and forwards the packet along the shortest path until it reaches its destination. Although this approach is simple and quite efficient, this routing pro-

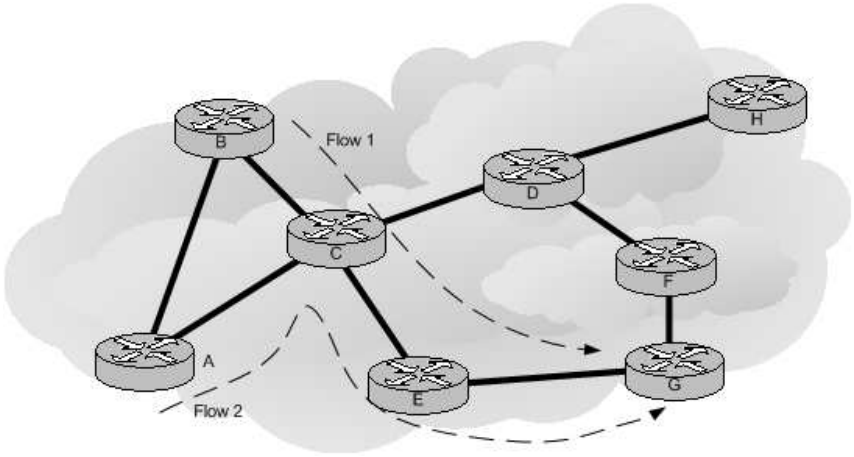

Figure 2.2: IP Routing Illustration

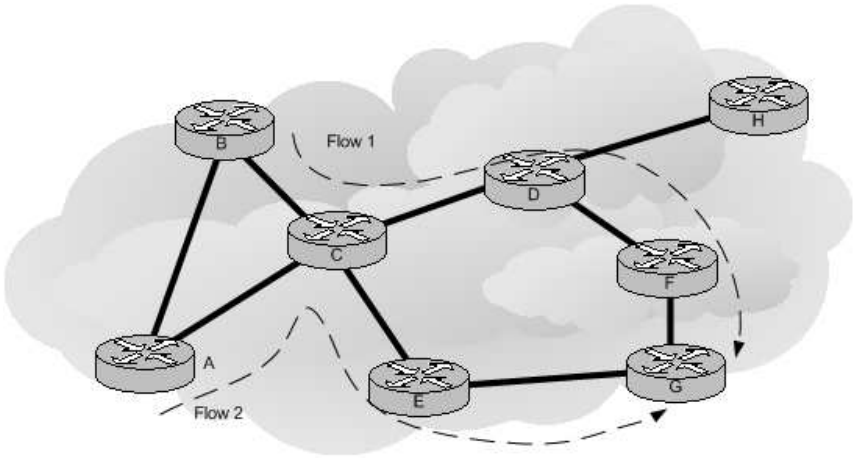

Figure 2.3: MPLS Switching Illustration

cedure imposes a major limitation on possible routing patterns as illustrated in 2.2. The two flows originating from router A and B destined for router G merge at router C and share the rest of the path to the destination. This might create congestion on the shared links, whereas other paths may be only lightly used or even empty. In 2.3, two LSPs are established; the first is B-C-D-F-G and the second is A-C-E-G. Flow 1 is associated with the first LSP and Flow 2 is associated with the second LSP. Even though both flows converge at router C, they are then split according to the predefined LSPs. This is an example

of MPLS routing flexibility, eliminating the dependency on destination-based routing.

2.3 Existing work in Traffic Engineering

This section provides a brief overview of a few prominent intra-domain traffic engineering methods.

2.3.1 Weight Setting Techniques

Weight setting techniques that do not require any protocol or router software modifications can be done in several ways. The idea of weight setting was initially proposed by Fortz and Thorup in [6]. The objective is to find a set of link weights such that there is an even load distribution of traffic across all links in the network by effectively penalising high link loads. There are two main contributions in their work:

This novel work utilises a heuristic method, known as Tabu search. Tabu search, one of many available local search techniques, enhances the performance of primitive local searches by storing solutions that have already been explored. Thus, repeated searches can be prevented. Local search algorithms require a movement description that defines how to explore the next solution from the current solution. There are two movements described here, namely a random single weight change and a load spreading across multiple paths. The first move involves increasing the weight of a heavily loaded link or to decrease the weight of a lightly loaded link. The second move requires the identification of OD-Pairs that are responsible for creating congestion in some parts of the network and adjusting link weights such that multiple paths with

equal length exist for these OD-Pairs. Once the solution is obtained, the solution is coded using a hash function for reducing the storage space required and enabling easier solution comparison. In the model, the objective is to minimise a cost function, which is an exponential queueing delay cost based on the link utilisation. In this thesis, the objective function is simplified into a piece-wise linear function to enable a linear programming problem representation.

The cost evaluation in the weight setting problem has proven to be computationally expensive and is identified as the bottleneck in the heuristic. This computation is not as simple as finding a shortest path for each OD-Pair, but it has to take into consideration the splitting of the load when multi-paths with equal length exist. In [6] [28], they proposed a method to compute the resulting load on all links in the network, given a weight system, efficiently. As it turns out, when the previous solution and the current solution are only slightly different, the cost function evaluation time can be improved as much as 85% using dynamic cost evaluation. It is desirable to do this because, if there is only one or just a few weight changes in the network, most of the routing patterns will not change (i.e. most of the flows are routed as before) with few exceptions, the dynamic cost evaluation will update these flows only.

In [12], Wang et al showed that it is possible to convert a set of routing patterns into a set of shortest paths, for which, in turn, shortest path metrics can be obtained (as long as the routing patterns do not have loops). The significance of this work is the identification of the dual variables in the multi-commodity flow problem formulation as the required shortest path metrics. However, the authors did not consider the uniqueness of the solution. It is true that a path to a particular destination can be converted into a shortest path, *but* this path may not be the only shortest path that exists to the des-

tination. This is important, because load balancing works based on multiple shortest paths.

In [11], Pioro formalised the weight setting problem in OSPF/IS-IS networks. The problem constraints are given as follows:

$$\sum_{p} x_{dp}(\boldsymbol{w}) = h_d \tag{2.1}$$

$$\sum_{d}\sum_{p} \delta_{edp} x_{dp}(\boldsymbol{w}) \leq \gamma_e c_e \tag{2.2}$$

$$\boldsymbol{w} \in \mathbf{W} \tag{2.3}$$

The constraints in eq. (2.1) enforces the demand requirements, which state that the total demand carried on all the paths associated with the demand d is equal to the total demand h_d. The second set of the constraints in eq. (2.2) restrict the induced load due to the weight set, $\boldsymbol{w}$, to be less than a specified fraction of the link capacity. The shortest path weight system is given in eq. (2.3). The unknowns in this problem are the values of the weights $\mathbf{W}$ and the amount of flow on the path p associated with demand d, x_{dp}. However, this formulation is not a true mathematical formulation, due to the dependency of the values of x on the weight system, $\boldsymbol{w}$. This dependency is due to shortest path routing and load balancing in OSPF/IS-IS networks. Hence, this formulation cannot be solved using mathematical tools, such as Linear Programming solvers. A mixed integer programming formulation of the shortest path routing allocation problem is also given in [11] based on A. Tomaszewski's contributions. However, this problem may be difficult to solve even for moderate sized networks, although this allows us to obtain exact solutions. The authors also contributed a different work wherein the objective looks at weight set-

ting under a variety of different network objectives [29]. Another LP-based contribution that also optimises the link delay is given in [30].

Following Fortz and Throup's work, researchers started to solve the problem using different local search techniques. A Genetic Algorithm is used in [31] [32] to find a weight set for given shortest path routing metrics. The OSPF weights are encoded in the string of zeros and ones which then form *individuals*. These individuals evolve according to reproduction rules, crossovers and mutations such that new individuals, which hopefully have a better fitness function, are created. The fitness function, in this case, is the cost function that penalises a high link utilisation as given in [6]. These iterations to create new individuals are carried out until a termination criteria is satisfied. Similar work with a different search algorithm, namely *simulated annealing* is given in [33].

In [34], Bley et al formulated a heuristic to find a set of shortest path metrics that result in single path routing. The context of the problem is a restoration problem, wherein for a case of a single link or node failure, a specific percentage of demand can be satisfied. The problem is formulated as a mixed integer linear programming problem. The authors then proposed a heuristic, that searches the neighbouring solutions based on the current solution. The neighbouring solutions are obtained by modifying the weights on under-loaded or over-loaded links.

In [35], Pioro et al present an OSPF routing optimisation problem, prove its NP-completeness and discuss possible heuristic approaches to solve it. A complete mixed integer linear programming formulation is described in this work. The problem is a flow allocation problem in a shortest path network. The unknown variables are flow variables and link weights. A proposed direct approach is to solve this MILP formulation by using a two phase approach.

The idea is to allocate all demands on a single path during the first phase and to try to find a weight system for which the paths realising demands are unique shortest paths. Although this approach is simpler than the direct approach, it may not yield a solution because the second phase does not always lead to a feasible solution. In this work, the experimental results for different networks are also given.

Guérin et. al. in [36], used the well known multi-commodity flow problem as the basis of the OSPF optimisation. The primal solution is used to find the value of the flow variables, whereas the dual variable is used to find the set of shortest path metrics. Unfortunately, the primal solution also produces arbitrary splits for the flow solution. Arbitrary splits are not supported in any current router's forwarding mechanism, nor in existing routing protocols. Generally, routers only support equal load balancing, through its Equal Cost Multi Path (ECMP) capability. This feature allows a flow to a particular destination to be split equally among paths that have equal cost to the destination. By manipulating the set of next hops for routing prefixes, the authors showed that it is possible to achieve near optimal traffic distribution (i.e. keeping the load below a specific fraction of the link capacity). Although this approach does not require any changes to the routing protocols, it does require changes in the control plane of the router to determine which next hops are allowed for a specific prefix. More details can be found in [36].

All the approaches described above require a knowledge of the network topology and the traffic demand for the global routing optimisation. Without this knowledge, it is impossible to carry out such tasks. In the event that only local knowledge is available, deflection routing can be utilised as a method to avoid congestion. In [7], the authors derived sufficient conditions that guarantee loop free forwarding. A node that detects congestion on one of its outgoing

interfaces can only deflect the traffic to one of its neighbours if the neighbour's distance to the destination is less that that of the node itself. This condition is sufficient to prevent a routing loop and the forwarding decision is still largely made based on the distance metric. Given that the network capacity is highly provisioned, traffic is deflected to an area where the network is not highly loaded. This makes deflection routing techniques effective in relieving congestion when there is local knowledge.

It is shown in [37] that having multiple paths to a destination might lead to bias splitting. Through measurement studies in a real network, it is concluded that even splitting can be off by 20% and it might replicate if the traffic meets multiple ties from source to destination. In this work, the authors showed that the solution without splitting is almost as good as the one with even splitting. The single path weight setting approach is done using tabu search as a search framework. Flow splitting is prevented by introducing a penalty on the cost function when a split occurs and increasing the the range of link weights. The authors presented the experimental results on real and synthetic networks and compared the result with different weight settings, notably to [6] and the Cisco inverse capacity metric.

2.3.2 MPLS Traffic Engineering

A framework of MPLS Traffic Engineering is given in [3] and summarised in [38]. The MPLS-TE framework requires the following functionalities:

Path management This component is concerned with the aspects related to selection of explicit routes and the instantiation and maintenance of LSP tunnels. Included in path management are path selection functionality (constraint-based routing), signalling protocols to establish and tear

down the LSP tunnels, such as RSVP-TE (Resource Reservation Protocol with Traffic Engineering extension) and the CR-LDP (Constraint based Routing Label Distribution Protocol).

Traffic assignment This component is concerned with all aspects related to the allocation of traffic to established LSP tunnels. It partitions ingress traffic (traffic coming into an MPLS domain) according to some policies and allots the partitioned traffic to established LSP tunnels - accordingly based on the requirement. Included here is how differentiated services behaviour aggregates are mapped onto LSP tunnels. This is one of MPLS features that distinguishes itself from ATM.

Network state information dissemination In order to carry out constraint based routing, additional information related to relevant topology state information, such as link delay and link residual bandwidth is required throughout the MPLS domain. This is accomplished by extending IGP to deliver additional information in the link state advertisement. Support for the OSPF protocol to carry this additional information is developed in OSPF with TE extensions.

Network management Using the MPLS approach to traffic engineer a network will only be successful if the network can be observed and controlled. This component includes a set of configuration, performance, accounting and fault management functions. This component also collects traffic statistics that can be used for statistical analysis and capacity planning purposes.

Most of MPLS Traffic engineering work is concerned with online optimisation of running networks. They propose algorithms to find routes for LSPs

as the request to build LSPs arrives. Upon completion of transmitting this information, an LSP will be torn down. This process is repeated all over again during normal network operation. The performance of these algorithms is evaluated based on the number of LSP requests that are accepted and rejected and the utilisation of the network. Examples of these approaches are [39], [40] and [41].

In [13], Gavish et al formulated the joint problem of selecting a primary path for each OD-Pair and a capacity value for each link in computer communication networks. The goal is to obtain the least costly feasible design where the costs include capacity and queueing components. The problem is then solved using Lagrangian relaxation and subgradient optimisation techniques. When the method was tested, they obtained good feasible and tight lower bounded solutions. However, this work did not target any specific routing technologies. It may require MPLS that allows laying down explicit routed LSP tunnels arbitrarily.

In [14], Wang et al used explicit routing algorithms for Internet traffic engineering because it is seen to be a much more capable solution for improving network utilisation than native IP routing. The optimisation problem considered in this paper is: *how to setup explicit routes to meet bandwidth demands between the edge nodes of the network and, at the same time, to optimise network performance*. The objective is to minimise congestion and maximise potential traffic growth. The formulation for single path and multi-path routing is presented as an LP and an MILP model respectively. Experimental results comparing the performance of these two methods against the widely used shortest-path and minimum hop algorithms are given.

2.3.3 Combined IP-MPLS Optimisation

Although MPLS technology is maturing at this stage; in practice, most existing OSPF/IS-IS networks will have a transition phase between pure IGP routing and MPLS. It is stipulated in [42] and references therein that the MPLS environment does not come without cost; configuration complexity and state space explosions might become issues in MPLS devices. Hence, it makes sense to take the MPLS traffic engineering feature and combine it with IGP simplicity to produce a state space reduction. In this work, the authors formulated an optimisation problem that minimises a combination of the maximum and average utilisation by finding the optimal set of paths. In addition to the capacity and transport constraints, the authors also introduced constraints that set a delay bound that governs the delay of the path and these bounds should not exceed a predefined value in order to support QoS. The optimisation model was then compared to the default OSPF and optimised MPLS networks.

The second part of the work is the development of the DDR (Decompose - Design - Re-assemble) algorithm to remove the bottleneck traffic from native IP networks and re-import it in an optimal and distributed manner into an MPLS overlay network. This heuristic identifies the flows that need to be rerouted and establish MPLS tunnels to re-route around the bottleneck.

In [43], it is also stated that the MPLS tunnels (LSPs) are only required if the routing paths given by a specific routing pattern cannot be realised by the IGP metrics. In this work, the authors formulated a number of routing optimisation problems including both single path and multi-path routing, with and without MPLS. However, this work did not present any experimental results.

In [44], Garcia et al are looking at the problem of LSP placement with

the objective of minimising the number of explicit routes and, at the same time, enforcing bandwidth utilisation and QoS delivery. Moreover, the model also provides optimal backup routes in the case of link failures in an MPLS network.

Similar work is given in [45] [46], [47] and [48].

MPLS has enabled the delivery of different forwarding behaviours depending on traffic requirements. In [49], Trimintzios et al introduced an architectural framework for integrated network management and control. They then considered the problem for long-term network dimensioning based on the requirements of contracted services and subsequent dynamic route and resource management in smaller time scales and varying network conditions. Unfortunately, this work did not provide a mathematical model or any experimental results, but merely some architectural descriptions.

2.4 Conclusions

This chapter has presented an overview of work discussed in the literature on traffic engineering to achieve route optimisation. In particular, it has provided the necessary background for understanding routing optimisation in native IP networks and emerging MPLS networks. This overview provides an appreciation of the problem domain in Internet Traffic Engineering. Furthermore, it also gives an indication of how routing problems in the Internet can be modelled and analysed mathematically. Although this list is not exhaustive - as it does not cover all the work done in the traffic engineering area - it provides enough information to gain an appreciation of the Internet traffic engineering approaches that have been taken so far, their relative advantages and disadvantages.

Chapter 3

LP-based Weight Setting for Online Traffic Engineering

We now consider the problem of optimising intra-domain routing patterns in IP-based networks. The optimisation process can be categorised into two separate notions, i.e. routing optimisation and routing adaptation. *Routing optimisation* is concerned with finding the global optimal routing solution irrespective of the current run-time network configuration. In most cases, this approach will give the best solution amongst all, however, it may require many changes to an existing network configuration. On the other hand, *Routing adaptation* aims to optimise the network and, at the same time, considers the current running configuration. It tries to find an optimum solution "close" to the existing routing pattern to avoid the negative impacts of routing instability when weight changes are invoked.

Weight setting approaches are used to traffic engineer an OSPF network. The idea is to find a set of link weights that perform well according to specified performance objectives. The weight setting problem has proven to be an NP complete problem. Heuristics are developed to find good solutions;

however, the run-times might turn out to be prohibitive for online traffic engineering applications. LP-based weight setting is a good candidate for online traffic engineering applications because it can compute the solution quickly. This chapter describes LP-based weight setting which is based on the classical multi-commodity flow problem. We investigate the solution quality based on topological properties.

3.1 Notation

Table 3.1 presented on the following page provides a detailed summary of the main notation that has been used throughout this thesis for the various proposed mathematical models.

3.2 Introduction

OSPF and IS-IS are the most commonly used routing protocols for routing traffic inside an AS. They belong to a group of link-state routing protocol classes, and they send Link State Advertisement (LSA) messages to their neighbours. These messages are then propagated to all of the routers in the AS. Ultimately, all the routers in the AS will have their link state database updated according to the received LSAs. Once this topology information is available, the router computes the next hop required to route packets on their way to the intended destination by using shortest path algorithms, such as Dijkstra's algorithm. Dijkstra's shortest path algorithm allows the router to build the shortest path tree from the router itself ("home node") to every other node recorded in the link state database. It then records the next hop (outgoing port or interface) so that it can attempt to deliver packets closer to their

Symbol	Explanation
$\delta_i^k(P)$	An indicator if link i is used in path P of OD-Pair k. Equal to 1 if it is used or 0 otherwise.
$\delta_i^{tk}(P)$	An indicator if link i is used in path P for traffic type t of OD-Pair k. Equal to 1 if it is used or 0 otherwise.
d^k	The amount of traffic demand for OD-Pair k.
d^{tk}	The amount of traffic demand for OD-Pair k for traffic type t.
u_i	The capacity of link i.
$f^k(P)$	A decision variable that denotes the amount of flow on path P of OD-Pair k.
$f^{tk}(P)$	A decision variable that denotes the amount of flow on path P of traffic t of OD-Pair k.
$c(P)$	The sum of link metrics or link costs along path P.
E	The set containing edges (links) in the network.
K	The set containing OD-Pairs in the network.
T	The set containing different traffic classes in the network.
P^k	A collection of paths belonging to OD-Pair k.
$a^k(P)$	A binary decision variable which takes the value 1 if path P is used to carry flow for OD-Pair k or 0 otherwise.
$a^{tk}(P)$	A binary decision variable which takes the value 1 if path P is used to carry flow for OD-Pair k or 0 otherwise.
c_i or w_i	Shortest path routing metric for link i.
R	The minimum residual link capacity in the network.
$h^{tk}(P)$	The hop count for path P that is used to carry traffic type t for OD-Pair k.
y_i	Additional capacity for link i that needs to be purchased to restore traffic above the required restoration level.
r^{tk}	The restoration percentage of traffic t of OD-Pair k.
p_i	The cost of purchasing additional capacity per unit flow on link i.
y_i	The amount of additional capacity needs to be purchased to accommodate demand on link i.

Table 3.1: Mathematical notation used throughout this thesis

destination.

In shortest path networks, all links have one or more associated routing metrics. In practice, only one of the routing metrics is used at a time. The most commonly used metric in OSPF or IS-IS is a “distance” metric, which means a link with a high metric value is less likely to be used compared to

links with a lower metric value. Hence, a link with a lower metric value is seen to be attractive for traffic. Traffic routing is then solely determined by the assigned link weights, without considering the link capacities. Loosely speaking, the traffic engineering process is required to ensure that none of the links is overloaded during normal network operation.

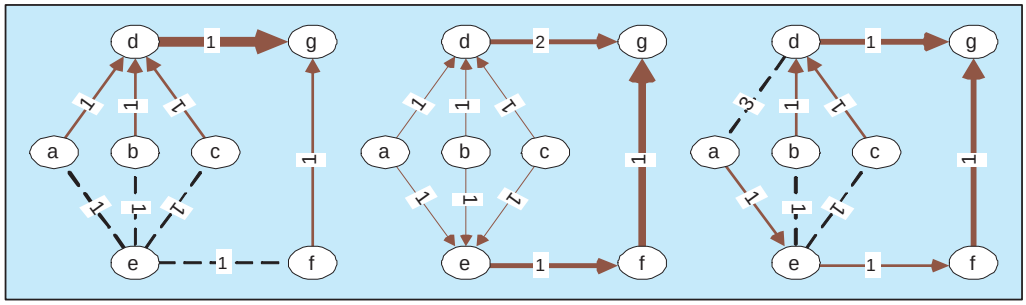

Figure 3.1: Weight Setting Example

An example of weight setting problem is given in Figure 3.1. The number on the links indicates the link metric. In these scenarios, node a, b, c and f sends one unit of demand to node g. The demands are sent according to their shortest paths. The resulting load on the links is indicated by the thickness of the arrows. All link capacities are assumed to be 2 units.

In the first scenario, link (d, g) is overloaded by 3 units of traffic (link utilisation = 150%). One fix could be increasing the metric on the congested link, i.e. link (d, g) from 1 to 2 as given in the second scenario. In so doing, there are equal cost paths from node a, b and c to node g, viz: via node d and via node e. The traffic from node a, b and c will be split evenly. The router does the even splitting through an Equal Cost Multi Path (ECMP) mechanism - provided that this has been enabled by the network administrator. However, the result of ECMP routing causes an overload on link (f, g) (giving link utilisation = 125%). In the third scenario, the metric on link (a, d) is modified to have the value 3. Hence, the shortest path for node a to node g is via node

e, whereas the shortest path for node b and node c is via node d. This is an optimal solution, where none of the link utilisations exceeds 100%.

It is often suggested that weight changes are "bad" because of the disturbance caused [28] to the network. We argue in the earlier section of this chapter that it is worthwhile executing weight changes, given that the network is congested. The time taken for all routers in a network to see the same network-wide view after an update message is called the convergence time. This study has been carried out using an additional module in the ns-2 simulator [50]. The second contribution of this chapter is a study that tries to measure the convergence time after a network event, such as a weight change or a link failure, takes place. Hence, we can postulate that, if the convergence time is a minimum then the disturbance to the network is also a minimum as well.

3.3 Fast Weight Setting for Handling Failures

Internet Service Providers (ISPs) fortify their backbone networks against failures by employing protection schemes or restoration schemes. Commonly implemented at the physical layer, a protection scheme involves the process of reserving network resources for backup paths. When the primary path fails, the traffic is automatically forwarded through a backup path. In contrast, a restoration scheme dynamically looks for a new path to the destination when a failure occurs by using IP routing protocols [51]. Why would we not use protection schemes? Protection schemes cannot protect against an IP router or forwarding software failure. A greater investment has to be made in improving the reliability of the equipment and provisioning spare capacity for use in the event of failures. By contrast, IP restoration schemes do not suffer from

any of these disadvantages, but they do offer a slower recovery time.

Sprint, a Tier 1 ISP and also one of the largest ISPs in the US, chose to adopt an IP-level restoration scheme [52] based on path recomputation through IP routing protocols, such as OSPF and IS-IS. With adequate capacity provisioning and careful network design, they concluded that IP restoration is a reasonably effective scheme for restoring on-going connectivity during failures in the Sprint network. Capacity provisioning in this network is achieved by maintaining the average utilisation of any link under 50% [52] [7].

However, even though their network is "sufficiently" over-provisioned, they report that IP restoration can cause a "hot spot". A hot spot is defined as a link carrying a load that is much more than its permissible traffic load. In a four month study they found that 80% of the time, hot spots are caused by link failures. The source of the link failure might be optical fibre cuts, router reboots, interface card failures, scheduled maintenance and software updates. It also showed that link failures happen on a daily basis [52].

In the event of hot spots, the network operator can change a single or a few link weights (metrics) to relieve the network. However, changing the link weights during failure is not recommended according to [53] [28]. The new weights have to be flooded throughout the network and each router has to recompute their shortest paths and update their routing table. During this period, packets might arrive out-of-order causing TCP back-off. The period of ensuing disruption could last for seconds and may take even longer before the network returns to equilibrium.

In [7], Iyer et al proposed the use of *deflection routing*. The idea is that a router deflects some of the traffic flow to neighbouring routers to alleviate congestion on one of its downstream links. The authors implied that the deflection process takes place within a single PoP (Point of Presence), in which

the topology is fully meshed. The deflection approach also requires that the intra-POP links be well over-provisioned. Furthermore, the implementation requires a modification of the router's forwarding plane.

An alternative and appealing approach is to prevent the hot spot forming in the first place – even when a link failure occurs. In [53] [54], the authors proposed a "resilient" or "robust" method based on a Tabu search heuristic, i.e. the impact of a link failure is reduced by choosing a judicious set of link weights. They have produced reasonably impressive results but there are some pitfalls in this approach, viz:

1. Their method only works for a single link failure. Exploring a set of weights that performs well with every combination of multiple link failures is computationally prohibitive. However, we note that multiple link failures may happen more often than expected since a single fibre cut often results in multiple IP link failures. A measurement study in the Sprint backbone network found that *multiple* link failures due to a fibre cut caused link loads to reach levels of 90% for 90 minutes [7]. The authors agreed that changing link weights would be appropriate. However, the current method of changing link weights based on heuristics is very slow in practice. Hence, they rejected this approach. Indeed, this is an example where our fast weight setting method offers a very attractive solution.

2. The "judicious" set of weights produces a small performance degradation in the absence of failure.

3. The models are based on the assumption that the equal cost multi-path (ECMP) facility splits the aggregate traffic to a particular destination equally. However, in [37], the authors reported measurement results

which indicate that there are systematic biases in even splitting of up to 20%. We estimate that this can cause relative variations in the "calculated" link utilisations by as much as 5-10%. Together with the degradation discussed in the previous items this, in turn, raises some questions about the effectiveness of these methods because to work well they need the link utilisations to be well specified.

We suggest changing the link weights if there are hot spots in the network. We agree that changing weights is not recommended during normal operation, because of the introduced disturbance. However, in the case of a single weight change, we argue that it is hardly a catastrophe. A single weight change only affects the flows that were using the congested link. For instance, if we have a link that is congested for, say, 30 - 40 minutes, every flow using that link will suffer inferior service. If we then calculate the required weight changes and update the link weight in the network, good service will be restored within tens of seconds.

This chapter deals with the issue of alleviating congestion caused by failures by using a fast weight setting method based on Linear Programming as described in [55] [10]. We envisage that our method is complementary to the robust weight setting methods. We present the theory of our LP method and performance results in different test scenarios. Using the splitting ratio concept, we also show how the initial solution affects the degree of solution uniqueness. Furthermore, we consider the performance of the method when we have uncertain information, such as inaccurate OD-pair traffic measurements.

3.4 Multi-commodity flow problem

This section describes the multi-commodity flow problem. We use the path flow formulation rather than the link flow formulation for easier representation and understanding. It should be noted that the shortest path flow allocation is actually an uncapacitated version of the multi-commodity flow problem (i.e. all constraints related to link capacities are ignored).

For each commodity k, let $\mathbf{P}^k$ denotes the collection of all directed paths from the source node s^k to the destination node t^k in the underlying network $\mathbf{G} = (\mathbf{N}, \mathbf{E})$, where $\mathbf{N}$ denotes a set of nodes and $\mathbf{E}$ denotes a set of links in the network. In the path flow formulation, each decision variable $f(P)$ is the flow on a path P for the k^{th} commodity. We define this decision variable for every directed path P in $\mathbf{P}^k$.

Let $\delta_i(P)$ be an link-path indicator variable, that is, $\delta_i(P) = 1$, if link $i \in P$ or 0 otherwise. Let $c(P) = \sum_{i \in E} c_i \delta_i(P) = \sum_{i \in P} c_i$ denotes the per unit cost of flow on the path $P \in \mathbf{P}^k$. Hence, we can formulate the multi-commodity flow problem as follows:

$$\text{Minimise} \sum_{k \in K} \sum_{P \in \mathbf{P}^k} c(P) f(P) \tag{3.1a}$$

subject to:

$$\sum_{k \in K} \sum_{P \in \mathbf{P}^k} \delta_i(P) f(P) \leq u_i \qquad \text{for all } \; i \in \mathbf{E} \tag{3.1b}$$

$$\sum_{P \in \mathbf{P}^k} f(P) = d^k \qquad \text{for all } \; k \in \mathbf{K} \tag{3.1c}$$

$$f(P) \geq 0 \qquad \text{for all } P \in \mathbf{P}^k \text{ and all } k \in \mathbf{K} \tag{3.1d}$$

The constraints given in (3.1b), are known as the bundle constraints, and they restrict the total of the aggregate flows to be less than the link capacity.

The constraints given in (3.1c) ensure that the total path flows for an OD-pair must equal the OD-pair demand. The non-negativity constraints given in (3.1d) ensure that none of the variables will be negative. The objective function (3.1a) tries to minimise the total cost of transferring flows over the network for every OD-pair.

3.4.1 Using a dual solution for modifying weights

Let σ^k be the dual multiplier associated with constraints (3.1c) and ω_i be the dual multiplier associated with constraints (3.1b). The dual of the above problem can be written as follows:

$$\text{Maximise} \sum_{k \in K} d^k \sigma^k - \sum_{i \in E} u_i \omega_i \tag{3.2a}$$

subject to

$$\sigma^k \leq \sum_{i \in E} \delta_i(P)(c_i + \omega_i) \qquad \text{for all } P \in \mathbf{P}^k \text{ and all } k \in \mathbf{K} \tag{3.2b}$$

Suppose that optimal dual multipliers are denoted by $\omega^* = (\omega_1^*, \omega_2^*, \ldots, \omega_E^*)$ and $\sigma^* = (\sigma^{*1}, \sigma^{*2}, \ldots, \sigma^{*K})$. Due to constraints given in eq. (3.2b), at optimality we have

$$\sigma^{*k} = \min_{P \in P^k} \sum_{i \in E} \delta_i(P)(c_i + \omega_i^*) \qquad \text{for all } \; k \in \mathbf{K}$$

and $\omega_i^* \geq 0$. Now, if we define the optimal link weights as

$$c_i^* = c_i + \omega_i^* \qquad \text{for all } \; i \in \mathbf{E}$$

then the link weight system, $c^* = (c_1^*, c_2^*, \ldots, c_E^*)$, has the property that all the

non-zero primal optimal flows can be realised only on the paths that are the shortest with respect to weights c^*.

3.4.2 The Complementary Slackness Conditions

We note that we shall have a dual multiplier for every constraint in the multi-commodity path flow formulation as defined previously. With respect to these dual multipliers, the reduced cost $c_P^{\sigma,\omega}$ for each path flow variable $f(P)$ is given by

$$c_P^{\sigma,\omega} = c(P) + \sum_{i \in E} \delta_i(P) w_i - \sigma^k$$

The reduced cost of a path for an OD-pair can be defined as the difference between its modified cost and the cost of the modified shortest path for that OD-pair.

The solution for the multi-commodity path flow formulation is optimal, if and only if it satisfies the path flow complementary slackness conditions. They are as follows:

$$w_i \left[\sum_{k \in K} \sum_{P \in \mathbf{P}^k} \delta_i(P) f(P) - u_i \right] = 0 \text{ for all } i \in \mathbf{E} \tag{3.3a}$$

$$c_P^{\sigma,\omega} \geq 0 \text{ for all } P \in \mathbf{P}^k \text{ and all } k \in \mathbf{K} \tag{3.3b}$$

$$c_P^{\sigma,\omega} f(P) = 0 \text{ for all } P \in \mathbf{P}^k \text{ and all } k \in \mathbf{K} \tag{3.3c}$$

The first condition (3.3a) states that the product of a link dual price and a link's remaining capacity is zero. This implies that a link dual price will be non-zero if and only if the link is over filled. The second condition (3.3b) states that the reduced cost of all path variable must be greater than or equal to zero. The modified shortest path(s) will have a reduced cost of zero, whereas

the remainder of the paths will have a non-zero reduced cost if their modified costs are not equal to that of the modified shortest path(s). The third condition (3.3c) states that the product of a reduced cost and a flow amount on a path is zero. The path carries a non-zero flow if and only if it is one of the modified shortest paths. However, not all modified shortest paths carry a non-zero flow. None of the modified non-shortest paths will carry any flow.

σ^k is the shortest modified cost path to send the commodity from node s^k to node t^k. Furthermore, in an optimal solution, every path from node s^k to node t^k that carries a non-zero flow must have the modified shortest path cost of σ^k.

3.4.3 The Revised Dual Simplex Method

We devised a revised dual simplex method based on the revised simplex method described in Taha's book [56]. The difference involves the condition for the selecting the entering and leaving variables and the starting point of the algorithm. It is not essential to be familiar with this method, because it ultimately gives the same solution as the revised simplex method.

The revised dual simplex method is given in the Appendix to this thesis.

3.4.4 A Working Example

The shortest path for every OD-pair provides the lowest cost among other paths to transfer the demand, hence the minimum cost can only be achieved if and only if every demand is routed on their shortest path. However, routing all the demand on the shortest path to achieve a minimum network cost might violate some of the bundle constraints. Hence, we cannot be sure that our initial solution is even feasible.

Consider Figure 3.2 which has one capacitated link and two demand flows.

The only link with the capacity restriction is the link between node 2 and node 5 ($u_{25} = 5$ units). The first demand, d_1, goes from node 1 to node 5 (3 units). The second demand, d_3, goes from node 2 to node 5 (6 units). We write out the multi-commodity path flow formulation for this problem and solve it using the revised dual simplex method. The numbers beside the links denote the metric / costs of the links.

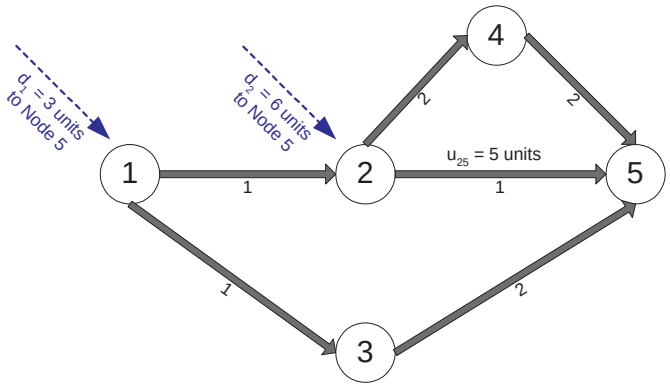

Figure 3.2: A 5 node network with capacitated links

LP Formulation

Let P_n^k be the n^{th} shortest path for commodity k. $c(P_n^k)$ is defined as $\sum_{(i,j)\in P_n^k} c_{ij}$.

To carry d_1, there are three possible paths, i.e.

P_1^1 traverses link (1,2), (2,5) $c(P_1^1) = 2$

P_2^1 traverses link (1,3), (3,5) $c(P_2^1) = 3$

P_3^1 traverses link (1,2), (2,4), (4,5) $c(P_2^1) = 5$

To carry d_2, there are also two possible paths, i.e.

P_1^2 traverses link (2,5) $c(P_1^2) = 1$

P_2^2 traverses link (2,4), (4,5) $c(P_2^2) = 4$

Writing out the equation as given (3.1) and putting them into the standard

LP form by adding a slack variable as follows:

$$\text{Minimise } z = 2f(P_1^1) + 1f(P_1^2) + 3f(P_2^1) + 5f(P_3^1) + 4f(P_2^2)$$

$$\begin{aligned} f(P_1^1) + f(P_1^2) + s_{25} &= 5 \\ f(P_1^1) + f(P_2^1) + f(P_3^1) &= 3 \\ f(P_2^1) + f(P_2^2) &= 6 \end{aligned}$$

or in tableau form

Basic	z	s_{25}	$f(P_1^1)$	$f(P_1^2)$	$f(P_2^1)$	$f(P_3^1)$	$f(P_2^2)$	Solution
z	1	0	-2	-1	-3	-5	-4	12
s_{25}	0	1	1	1	0	0	0	5
$f(P_1^1)$	0	0	1	0	1	1	0	3
$f(P_1^2)$	0	0	0	1	0	0	1	6

$\underbrace{\qquad\qquad}_{basic}$ $\underbrace{\qquad\qquad}_{nonbasic}$

The current basis $\mathbf{B}_0$, which contains all basic variables, yields an infeasible solution[1], i.e.

$$\mathbf{X}_b = \mathbf{B}_0^{-1}\mathbf{b}$$

$$\begin{pmatrix} s_{25} \\ f(P_1^1) \\ f(P_1^2) \end{pmatrix} = \begin{pmatrix} 1 & -1 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 5 \\ 3 \\ 6 \end{pmatrix} = \begin{pmatrix} -4 \\ 3 \\ 6 \end{pmatrix}$$

Iteration 0

For convenience, we shall solve the problem using a matrix format. Accord-

[1]The value of $\mathbf{B}_0^{-1}$ is calculated using the approach described in Appendix

ing to the dual feasibility condition for the dual simplex method, the leaving variable is the basic variable having the most negative value. Our leaving variable is s_{25}.

The entering variable is the non-basic variable that has

$$\min_{\text{Nonbasic } x_j} \left\{ \left| \frac{z_j - c_j}{\alpha_{rj}} \right|, \alpha_{rj} < 0 \right\}$$

where α_{rj} is the constraint coefficient of the tableau associated with the row of the leaving variable x_r and the column of the entering variable x_j. However, the value of α_{rj} is not readily available. It can be calculated using

$$\alpha_{rj} = (\mathbf{B}^{-1}\mathbf{P}_j)_r$$

The value of $z_j - c_j$ is given by $\mathbf{C}_B\mathbf{B}^{-1}\mathbf{P}_j - c_j$. Hence, for $f(P_2^1)$, the value of $z_j - c_j$ is

$$\begin{aligned} z_j - c_j &= \begin{pmatrix} 0 & -2 & -1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} - (-3) \\ &= 1 \end{aligned}$$

Similarly, we can calculate the values of $z_j - c_j$ for $f(P_3^1)$ and $f(P_2^2)$ which are 3 and 4, respectively.

Variable	$f(P_2^1)$	$f(P_3^1)$	$f(P_2^2)$
$(z_j - c_j)$	1	3	4
α_{1j}	-1	-1	-1
Ratio $\left\lvert \frac{z_j - c_j}{\alpha_{1j}} \right\rvert$	1	3	4

The ratios show that $f(P_2^1)$ is the entering variable.

The dual variables, $\mathbf{Y}$, can be calculated as follows $\mathbf{Y} = \mathbf{C}_B\mathbf{B}^{-1}$. Hence, at this iteration, the value of $\mathbf{Y}$ is

$$\mathbf{Y} = \begin{pmatrix} 0 & -2 & -1 \end{pmatrix} \begin{pmatrix} 1 & -1 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & -2 & -1 \end{pmatrix}$$

The dual variable for link (2,5) is zero. The modified costs for the shortest path to carry d_1 and d_2 are 2 and 1, respectively.

We repeated the calculations for a further two iterations. The solution at the completion of iteration 2 is given by

$$\begin{aligned} X_{B_2} &= \mathbf{B}_2^{-1}\mathbf{b} \\ \begin{pmatrix} f(P_2^1) \\ f(P_2^2) \\ f(P_1^2) \end{pmatrix} &= \begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 5 \\ 3 \\ 6 \end{pmatrix} = \begin{pmatrix} 3 \\ 1 \\ 5 \end{pmatrix} \end{aligned}$$

A set of dual variables $\mathbf{Y}$ is given by

$$\begin{aligned} \mathbf{Y} &= \mathbf{C}_B\mathbf{B}^{-1} \\ &= \begin{pmatrix} -3 & -4 & -1 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \\ &= \begin{pmatrix} 3 & -3 & -4 \end{pmatrix} \end{aligned}$$

The optimal objective function is given by

$$\begin{aligned} z &= \mathbf{C}_B\mathbf{X}_B \\ &= \begin{pmatrix} 3 & 4 & 1 \end{pmatrix} \begin{pmatrix} 3 \\ 1 \\ 5 \end{pmatrix} \\ &= 18 \end{aligned}$$

The primal LP solution sends d_1 via P_2^1 and sends d_2 via P_1^2 and P_2^2. The dual LP solution gives the dual variable for ω_{25}, σ^1 and σ^2. The modified cost of a path is given by $\sum_{i \in P} \omega_i + c(P)$. The costs of the modified shortest paths for d_1 and d_2 are 3 and 4, respectively. P_1^1 is no longer carrying any traffic because its modified cost is higher than that of P_2^1. P_1^2 and P_2^2 have the same modified cost path, so both of them could carry traffic, the magnitude of this traffic is specified by the primal solution.

3.5 Splitting Ratios concept and approximation

Routers make their routing decisions independently, based on shortest path calculations which are, in turn, based on a set of link weights (costs) for the whole network. From the results in Section 3.4.1, we can see that if the modified costs, $c_i + \omega_i$, were used as the OSPF weights, then a traffic demand would automatically be routed on a path of cost σ^k. For demands where only one shortest cost path exists, the actual routing solution is identical to the primal LP solution. Setting the OSPF weights to the modified costs will also initiate cases in which demands have multiple shortest paths of cost σ^k. For these cases, the actual routing solution is more complicated and requires the

calculation of node splitting ratios.

In practice, each router has a forwarding table in which each entry contains a destination and the link(s) that a packet should take to reach that destination. To implement the primal solution from the previous section in a real network, when there is more than one shortest "modified cost" path available from a particular node to a particular destination, a splitting ratio must be calculated.

A splitting ratio is defined as the proportion of incoming aggregate demands going to a particular destination across different outgoing links. Using the primal solution we find that for node n the splitting ratio for traffic destined for node t on link (n, j) is given by

$$\begin{aligned} R_{nj}^{nt} &= \frac{\text{incoming aggregate traffic destined to } t \text{ going to node } j}{\text{total incoming aggregate traffic destined to } t} \\ &= \frac{\sum_{k \in D_t} \sum_{P \in P_{sh}^k} \delta_i(P) f(P)}{\sum_{l \in A_n} \sum_{k \in D_t} \sum_{P \in P_{sh}} \delta_i(P) f(P)} \end{aligned}$$

for $(n, j) \in A_n$, where A_n is the set of all outgoing links at node n and D_t is the set of flows going to t that pass through node n. We note here that P_{sh}^k is the set of shortest "modified cost" paths that each flow can take to destination node t from node n and the general boolean operator is equal to one if a link i lies on a path P and zero otherwise. The splitting ratios are calculated and distributed to each node. Unfortunately, there are no commercial routers that support this method. One of the reasons for this is the need for intensive computation due to floating point calculations in order to implement the exact splitting ratio in the router. Furthermore, there should also be a mechanism to distribute the splitting ratio information to routers and the current routing protocols do not support this.

Implications from the LP Solution to OSPF Routing

We note that the splitting ratio can take any floating point values between 0 and 1, which means that if we were to implement this in real networks, we would need the router to perform unequal traffic splitting based on the splitting ratio value. However, at the present time, routers generally only support equal splitting. This means that if there are two equal cost paths to a destination, then the traffic to the destination will be split into two halves and sent equally on both paths. This even splitting feature is called Equal Cost Multi Path (ECMP) and is a feature of standard OSPF implementations.

Different schemes to approximate splitting ratios are given in [57]. A scheme that is readily implementable in OSPF networks approximates the unequal splits by splitting them equally among the outgoing links. The disparity between the LP solution and the implementation in true OSPF networks is caused by this approximation. Although this approximation is very crude, given the small number of splitting ratios that will be required, the approximation is still useful.

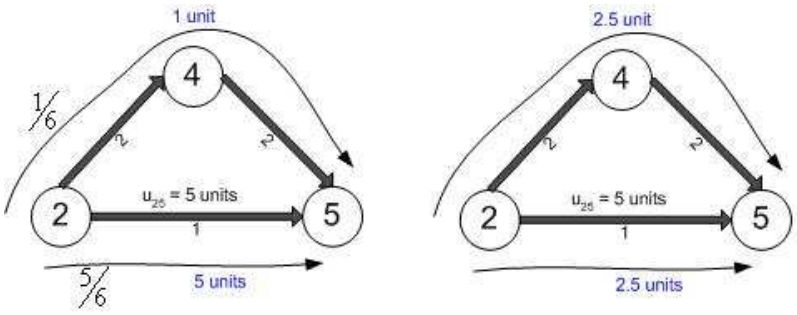

Figure 3.3: Illustration of the splitting ratio

To illustrate this, we take the previous example from Section 3.4.4. The part of the LP solution to this problem is given by the left-hand figure of Fig. 3.3. Having approximated the splitting ratio equally, we have the result in the

right-hand figure, which dictates that the flow be split into 2.5 units each. In this example, it is obvious that the approximation gives better load balancing than the LP solution.

3.6 Experimental Results

3.6.1 Experimental Setup

We tested our method on networks made up from "standard" AT&T networks joined together. The first network has 56 nodes and 200 unidirectional links. The network is given in Figure 3.4. The second has 90 nodes and 314 unidirectional links. Thus, assuming a traffic flow between each node, there are 3080 flows for a 56 node network and 8010 flows for a 90 node network. Each distribution of link capacities is shown in the legend underneath Table 3.2. The traffic matrices were generated randomly using a uniform distribution, the total throughput being just large enough to heavily overload the network but still being feasible.

3.6.2 Results and Discussions

We started the experiment by running the 56 node network under different starting sets of weights and different link capacity bounds. There are 3 different starting weights, namely InvCap (inverse link capacity), Unit (hop count) and Rand (random weight assignment). There are also 3 bounds on the link capacity that has been used; namely, S, M and D. The bounds are given underneath Table 3.2.

We load the network such that the LP optimal solution gives a 100% link load for each of experiments. Then we use the starting weight set (Initial

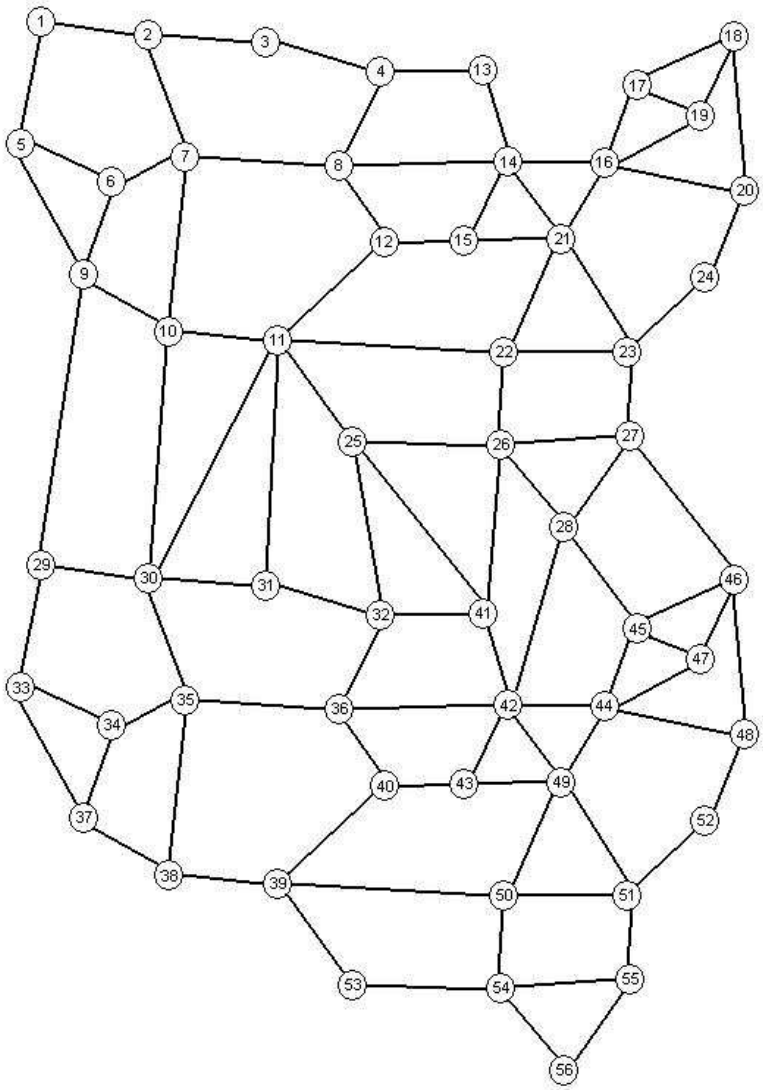

Figure 3.4: 56-node test network

Cost Type column) to compute the shortest path maximum utilisation. We then optimise the network using LP-based weight setting, apply the weight changes and recompute the maximum link utilisation. The new utilisation is given in the LP Max Util column. We also include statistics for the number of weight changes and the number of splitting ratios. For example, if we look at the first case (S, InvCap). Having the inverse capacity as the starting set of weights results in having a shortest path maximum utilisation of 243.6%.

Link capacity	Initial Cost Type	Shortest Path Max Util	LP Max Util	Number of split ratios	Number of weight changes
S	InvCap	243.6%	126.0%	178	24
M	InvCap	266.3%	134.1%	247	27
D	InvCap	204.3%	118.2%	204	16
S	Unit	229.0%	185.2%	1052	9
M	Unit	331.3%	149.1%	1019	17
D	Unit	372.5%	158.1%	1024	14
S	Rand	270.5%	146.3%	204	32
M	Rand	361.2%	112.4%	217	31
D	Rand	552.9%	116.3%	298	27

Table 3.2: Utilisation reduction with LP-based weight setting on the 56 node network
Legend: S - 400-600 units M - 200-800 units D - 100-900 units

After optimising the network and applying 24 weight changes, we reduced the maximum utilisation to 126%. The same applies for the other 8 experiments.

It is important to notice here that having hop count (Unit) as the starting weight does not yield good results. The results differ by at least 50% when compared with the LP optimal. In these cases, the numbers of splitting ratios are in the order of one thousand. These indicate there are very many equal cost paths and each of these has an arbitrary proportion split. Hence, the number of splitting ratios and the maximum link utilisation after optimisation can be indicators of how far the routing solution differs from the LP optimum solution.

The reason for running the example with many different sets of link capacity bounds is to see how bad is the effect of approximating unequal splitting ratios by taking equal proportions. Consider a single node that has a number of splitting ratios and two outgoing links having different capacities. By varying this link capacity, we would be able to observe the effect of excessive loads due to equal splitting on the smaller link.

The results of the experiment on a 90 node network are given in Table 3.3. This time, only the S category of link capacity bounds is used. The results show that the starting link weight affect the number of splitting ratios and the resulting maximum link utilisation after LP-based optimisation is used. These results also confirm the conclusions obtained from experiments made on the 56 node network.

Link capacity	Initial Cost Type	Shortest Path Max Util	LP Max Util	Number of split ratios	Number of weight changes
S	InvCap	355.3%	119.7%	565	38
S	Unit	262.1%	152.3%	2536	31
S	Rand	295.6%	113.6%	730	49

Table 3.3: Utilisation reduction with LP-based weight setting on the 56 node network
Legend: S - 400-600 units

3.6.3 Limiting the number of weight changes

We introduce a new parameter α into the bundle constraints 3.1b, such that they become

$$\sum_{k \in K} \sum_{P \in \mathbf{P}^k} \delta_i(P) f(P) \leq \alpha u_i \qquad \text{for all } i \in \mathbf{E}$$

By varying the capacity limit using a scaling parameter α, a limited number of weight changes can be attained. Intuitively, it works as follows: when the capacity scaling factor is very large, the shortest path solution is feasible. As the scaling factor is reduced, the shortest path solution is no longer feasible, because one of the links carries more traffic than its scaled capacity. Once the LP is solved, the first non-zero dual price will appear on that link. Reducing the scale factor further will cause the second link to overload. This

link will then have a non-zero dual price, and so forth. This is how we can control a limited number of weight changes.

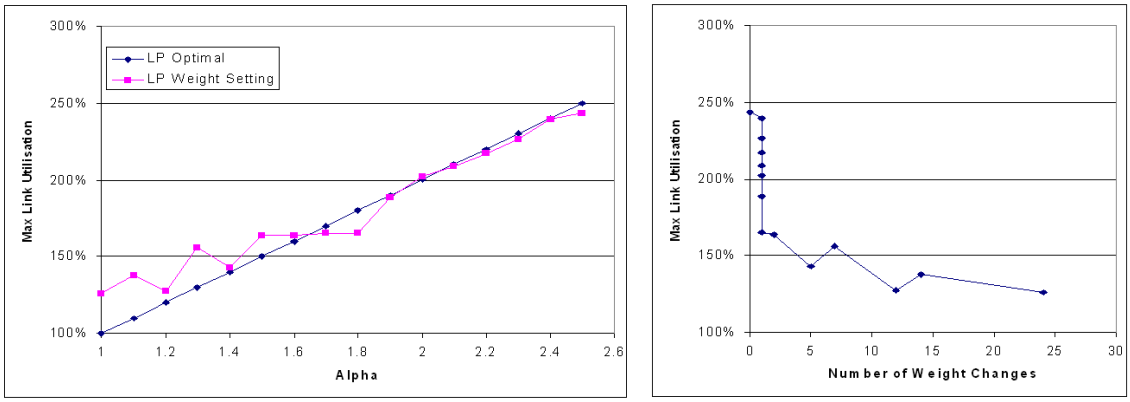

Figure 3.5: Deviations from optimality and varying the number of weight changes

The left subfigure of Figure 3.5 shows how far the LP-based weight setting solution deviates from the LP optimal as we reduce the value of α. When the value of α is still sufficiently large, the LP weight setting solution follows the optimal quite closely (because there are only a few splitting ratios here). The right subfigure of Figure 3.5 shows the reduction of the maximum utilisation as we execute varying numbers of weight changes. It is noted that for this problem, with just a single weight change, we can reduce the maximum utilisation by 80%. Thus, this shows LP-based weight setting is the equivalent of an online "hot-spot" (severe network congestion) removal application.

We then "engineered" the network, ensuring that every link has the same amount of spare capacity (slack) to handle the re-routed traffic. Following the example of Sprint, we set our overload limit to 50%. Thus allocating a slack of 200 units on every link corresponds to running the network at an average of 30% utilisation. This is chosen because Sprint run their backbone network between 20% to 25% utilisation as indicated in [52]. We also tested

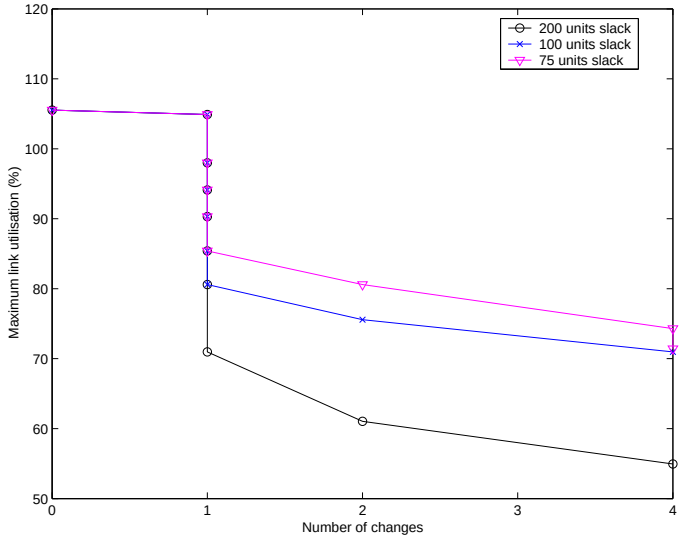

Figure 3.6: Hot Spot Utilisation Reduction in the 56 node network with a few weight changes

our method on a network with smaller amounts of slack (100 units and 75 units). These correspond to running all links at an average of 40% and 42.5% utilisation (very close to the overload limit). We show that our method works even when there is not much spare capacity to re-route the flows.

Generating a single hot spot in a simulated environment is not as easy as we thought. In our experiment, we found that a single link failure is not sufficient to cause a severe hot spot (as noted in [52] - 90% utilisation). We could generate a hot spot by bringing down multiple links. However, this results in a hard combinatorial problem. Hence, we resorted to increasing the size of the flows that use a particular link proportionally until the link utilisation is as high as the one in noted for the Sprint network. In Fig. 3.6, we have a single hot spot whose utilisation is as high as 105%. Note that this is a very large hot spot; the link has to carry about twice the traffic load compared to its permissible capacity. With a single weight change, our method can reduce the hot spot utilisation to 70% (200 units of slack). When the network does not

have so much slack, it is more difficult to re-route the traffic away from the hot spot. A single weight change reduces the hot spot utilisation to 80% and 95% for 100 and 75 units of slack, respectively. However, allowing a few more weight changes helps to bring the hot spot utilisation down to 55% of the 200 units of slack. In the network with smaller slack values, we also managed to reduce the utilisation to around 70%.

Calculation Time

Our method consists of two processes. The first is to calculate paths for all OD-pairs. The second is to solve the LP problem and to obtain the dual prices. We calculate a set of paths when the network is operating normally. Calculating paths for all OD-pairs takes 30 seconds for our 56 node network. The second process takes at most 3 seconds. Therefore, the total time to obtain a new set of link weights is below 35 seconds for this particular network. The results for a bigger network, 90 node 314 unidirectional link network is also given in Table 3.4.

Process	56 node network	90 node network
Path calculation for all OD-Pairs	30	324
LP formulation and solving	3	31

Table 3.4: Solution calculation time (seconds)

3.7 Convergence in OSPF

In networks running a link state protocol, such as OSPF or ISIS, weight changes are considered bad because of the disturbances caused to the network. From the network point of view, extra information has to be flooded

throughout the network to announce changes to the information about a link. Although the amount of the extra information is insignificant, it may increase the amount of traffic on the network. From the end customers' point of view, the disturbance shows in a brief reduction of download speed or an increased delay for a short period of time in IP telephony.

After all the routers in the network have received notifications about a weight change, they need to recompute their routing tables. Depending on the size of the network and the shortest path computation algorithm used, this process consumes a just a small fraction of time in the router's CPU. Depending on the result of this computation, the next hop forwarding information may be changed for some destination prefixes. In this case, packets belonging to the affected flows will be forwarded on a different path.

Depending on the queueing time and the propagation delay between the original path and the new path, it could be found that packets may be arriving out-of-order at the destination node. Whilst TCP is capable of packet re-ordering, it misinterprets out-of-order delivery as packet loss. The sender retransmits, although no actual loss has occurred, resulting in throughput degradation. However, the duration of the packets arriving out-of-order can be very short, hence the impact on the throughput reduction may not be so severe.

In this work, routing convergence is defined as the time required for all the routers in the network to have the same view of the forwarding information. In the event of a link failure, the convergence time comprises the following:

1. Detection of an interface being down or up.

2. Initial delay before the protocol stack is notified of the link status change. This is required to filter out any link transient times that may last only

for a few milliseconds.

3. Initial delay to generate a new LSP that informs other routers about the event. The rate of LSP flooding needs to be limited in a large network because of the possibility of excessive numbers of LSP packets (lspThrottle and lspDelay).

4. LSP flooding across the network. The flooding time is dependent on the network architecture and the link propagation time.

5. Delay between arrival of an LSP and the start of SPF computation. The period is used to collect a few LSPs together and just do the SPF computation once (SPFDelay). The SPFDelay will play an important part in our analysis.

6. SPF computation process and update to routing information base. The incremental SPF algorithm has been developed. It is reported that this can speed up the computation process by around 85% [58].

7. Updates from routing information base to forwarding information base in the line card.

In [58], the authors showed how to improve the routing convergence time. They proposed an approach to reduce the default pacing timers in the routing protocol, which can be done quite easily. Other approaches that have been presented involve modifying the router architecture in hardware and/or at the software level. The objective is to reduce the time for forwarding information unavailability.

Although the convergence process due to a link failure and a weight change looks similar, we argue that the disturbance during the convergence process

due to a weight change is much less than that of a link failure. The reason is as follows: In the event of a link failure, the forwarding information to some prefixes is lost once the link is down. The router does not know how to reach the destination until all the 7 steps outlined above are completed. Only after the 7^{th} step is completed, the router knows about the new forwarding information for the affected destination prefixes. Although, a weight change requires the routers to do SPF computation, the forwarding information is not lost. The router still can use the old forwarding information that resides in the forwarding table during steps 3-6. Note that step 1 and 2 are not relevant in the event of weight change. The disturbance caused by the weight change is mainly due to steps 6 and 7. Step 6, the SPF computation, increases the router load for a brief period. While step 7 is being carried out, the router may perform incorrect forwarding due to inconsistency in the route processor and its forwarding table.

We used ns-2.1b8 patched with the IS-IS routing module [59] to carry out the simulation work. The simulation is used to model the convergence process from steps 3 to step 6. Nevertheless, steps 1 & 2 only add a delay by just a few milliseconds. Simulating step 7 is not feasible since this step is dependent on the architecture of the router. In this work, we assume that multiple weight changes can be carried out and executed simultaneously in the network.

3.7.1 Convergence Time

LSP flooding time in a network is dependent on two factors, namely the queueing delay and the link transmission time. Queueing delay is the time that a packet spends within an interface queue in order to be served or forwarded to the next hop router. Link transmission delay is the transmission line propaga-

tion time due to electrical or optical properties of the transmission materials (delay in layer 1 of the OSI model).

To observe the queueing delay of the control packets / LSP in the ns-2 test network, bursty traffic was required to be flooded because packets are queued during the burst period, and those are released in the idle period. A traffic demand was randomly generated from a uniform distribution, and each flow of the traffic demand was used as a mean arrival rate from a Poisson traffic source to simulate bursty traffic.

Theoretically, the average delay of a packet in a queue is proportional to the arrival rate of the traffic. Therefore, the queueing delay of the control packet is expected to increase as the arrival rate of traffic increases. However, due to the characteristics of the burst traffic, it does not guarantee that the proportional relationship between the queueing delay of the control packet and the arrival rate always holds. In other words, although the arrival rate is high, if the control packets are flooded through the network during the idle period, the control packets will experience less delay. That is why our observation suggested that the convergence time tends to increase with some fluctuations as we increase the traffic arrival rate.

Another interesting observation is the relationship between the convergence time and the transmission delay of the control packets. The transmission time is defined as the time required propagating a packet from one source to end of the medium. For instance, if a packet travels across multiple links, the transmission time experienced by the packet can be obtained by adding the transmission time spent in each link. It means that the total transmission time of a packet increases as the packet travels across more links.

When a weight change occurs on a link, control packets are flooded throughout the entire network to synchronise this new information. Let's consider two

different cases where weight changes occur at the centre and at the edge of a network. The maximum number of links that a control packet floods in the former case is likely to be less than the number of links in the latter case. In other words, the convergence time is less when weight changes take place in the centre of the network. The example shows that the actual location of the weight changes affects the protocol convergence time.

There is a common belief that multiple weight changes disturb the network more than a single weight change. This is because multiple weight changes generate more control packets, and it takes more time for them to be flooded. It may be true in the sense of the queueing delay (Although it only causes a slight increase the queueing delay), however, it is not true in the sense of transmission time. The completion time of flooding all control packets caused by multiple weight changes is the same as the completion time of flooding the control packet which travels the most number of links (or the radius of the network). It implies that multiple weight changes do not necessarily increase the transmission time. Therefore, the impact in terms of transmission time of multiple weight changes might be the same (or even less) than a single weight change in an extreme scenario (Imagine when multiple weight changes occur in the centre of a network, and a single weight change occurs at the edge of the network). Table 3.5 shows the convergence time as the number of weight changes invoked in different parts of the 56-node network.

3.7.2 Throughput reduction during the convergence

Without a doubt, weight changes disturb the network because control packets containing new network state information are flooded throughout the entire network, and based on the new information, traffic needs to be re-routed. It

Weight changes and location	Flooding time (ms)
A weight change on the edge	25.244
2 weight changes on the edge	25.244
3 weight changes on the edge	25.244
A weight change on the centre	15.178
2 weight changes on the centre	20.237
3 weight changes on the centre	20.237

Table 3.5: Flooding times due to weight changes

may cause problems with packets out-of-order, loss, delay, and degradation of network performance. The question might be: how much disturbance does the network experience during the period. To provide a possible answer for this question, the performances of TCP and UDP traffic streams were observed during weight change period in our ns simulation study. Figure 3.8 and 3.7 show the variation of the throughput and the sending rate during the weight change period with TCP and UDP traffic sources respectively.

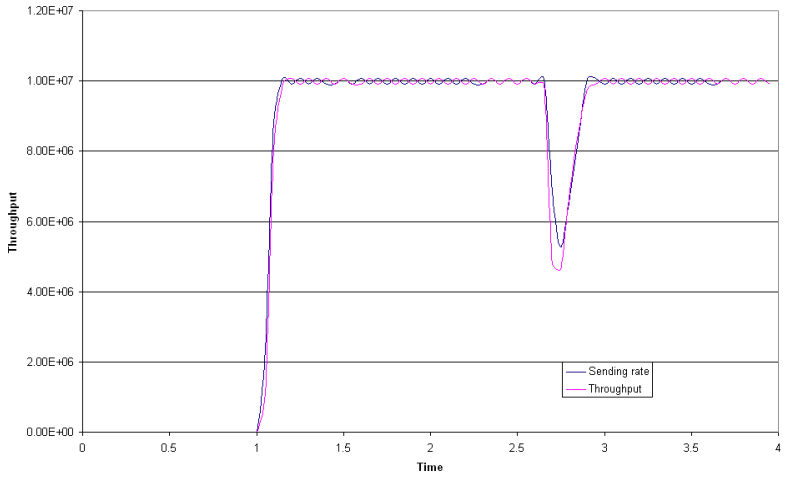

Figure 3.7: Reduction in TCP sending rate and throughput

In the TCP case, both the throughput and the sending rate are reduced during the convergence period, because TCP detects the delay and loss ac-

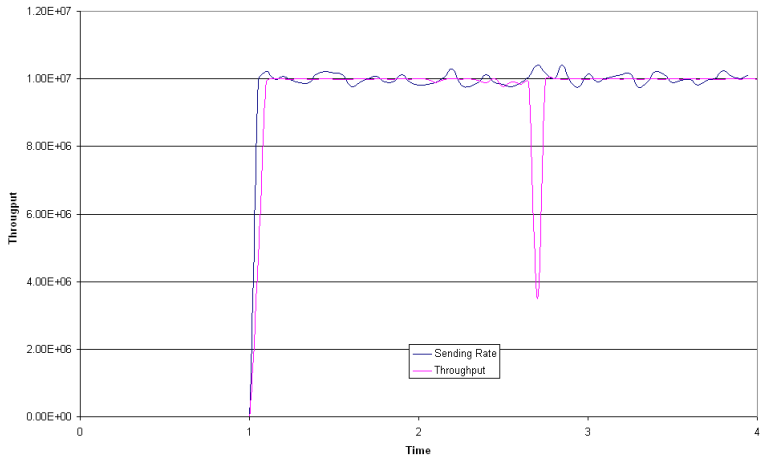

Figure 3.8: Reduction in UDP sending rate and throughput

knowledgement packets from the receiver during the disturbance period, and reduces the window size, which causes the throughput and the sending rate to be decreased. That is why the throughput and the sending rate are dropped from 2 to 2.5 seconds. TCP has a function to re-transmit lost packets so that the packet loss problem is not significant for TCP traffic.

On the other hand, UDP traffic keeps sending traffic with the same rate because UDP does not control the rate in terms of network state. However, the UDP traffic also experiences throughput drops in the same way as TCP traffic. This is because some UDP packets are lost and delayed while traffic is re-routed based on the new weights.

3.8 Conclusions

In this chapter, we showed that the multi-commodity flow problem can be utilised as the basis for shortest path network optimisation. In particular, by using the dual variables as weight modifiers, it was shown that new shortest paths can be created to reroute traffic away from the congested links. Fur-

thermore, we show that LP-based weight setting optimisation, although the solution is not unique, is a good candidate for an online traffic engineering optimisation due to its computation speed and its feature of reducing utilisation on the most congested link. In the second part of this chapter, we analysed the components of convergence time and hence we can now estimate how long it will take for a network to converge when weight changes are invoked. We found out that the convergence time, due to weight changes, is determined by the location of weight changes. The weight change itself does not have a catastrophic effect on the network; however, it does cause the reduction in throughput for a brief period as the experiment suggested.

Chapter 4

MPLS Optimisation

MultiProtocol Label Switching (MPLS) provides a framework for flexible traffic engineering via its constrained shortest path computation or explicit routing capability. In this chapter, MPLS routing models with two different objectives that utilise MPLS explicit routing are presented and discussed. The models are extensions of classical multi-commodity flow problem. The objectives of the two models are to minimise the network cost and maximise the minimum residual link capacity. Using the ns-2 simulator, the model that *maximises* the minimum residual link capacity is proven to deliver better performance in terms of network throughput and packet loss. However, this formulation results in a Mixed Integer Linear Programming (MILP) model that is hard to solve as it requires substantial computational effort (and thus taking a longer time to solve). Given that the models are intended for network planning, then the timing is not the issue but network size increases the state space.

Establishing explicit route Label Switched Paths (LSPs) may be seen as an expensive practice in terms of network setup and maintenance as the network size grows larger. Hence, an optimisation scheme that minimises the

number of explicitly routed LSPs is desirable. In the second part of this chapter, we show that it is possible to construct LSPs based on appropriate IGP metrics, such that the number of explicitly routed LSPs can be minimised. Furthermore, this approach is much more faster than the MPLS path selection method that was presented earlier. This gives the capability to react rapidly should it be required to react to network congestion.

4.1 Introduction

Multiprotocol Label Switching (MPLS) [8] was originally developed to achieve a higher switching speed. The idea behind MPLS is to affix a short fixed length label to IP packets when packets enter an MPLS domain. Packets with the same MPLS label indicate that they belong to the same Forward Equivalence Class (FEC). Packets in the same FEC will receive the same forwarding treatment. Hence, packets with the same label will be forwarded along the same path to their destination. This label is used to rapidly guide the packet through a pre-defined tunnel, which is known as a Label Switched Path (LSP).

However, the MPLS's switching speed benefit is slowly diminishing because advances in processing power for native IP routers has enabled the IP table look-up procedure to become much "faster". Nowadays, MPLS technology is used because it offers many other advantages, namely traffic shaping and policing, class-based routing, traffic monitoring, and most importantly, a framework for traffic engineering [8] [3] [9].

With an increasing number of networks supporting MPLS features, it has become mandatory to use this technology to perform traffic engineering. It is also shown in [28] that MPLS can be used to carry out optimal routing, wherein the objective is to keep the link utilisation for all links below a certain

target value. The most important feature of MPLS in the context of this work is its ability to perform TE using its explicit routing capability. Given an MPLS enabled domain, a network operator can control how traffic flows are routed in their network with different granularities.

In this work, we are focussing on off-line calculation. Given a traffic demand matrix and a network topology, the question to be addressed is: *which path should be chosen for each individual OD-Pair to comply with the performance objective over the long term?* Two mathematical formulations are developed and discussed in this chapter to address this problem. The problem of path selection using a simple performance objective has already been discussed in [60]. However, this work did not specifically target MPLS technology. This chapter also presents simulation results to verify our methods.

However, MPLS optimisation issues do not finish here. The administrative cost in terms of setting up explicitly routed LSPs may be prohibitive for some network administrators, due to the size of the network or the number of OD-Pairs. Hence, it is much easier to build most of the LSPs based on available routing information such as IGP routing metrics and only explicitly route a smaller number of LSPs. In the second part of this chapter, the LP formulation from the previous chapter can be used to determine which LSPs need to be explicitly routed.

The principal contributions made in this chapter can be summarised as follows: Firstly, the Mixed Integer Linear Programming (MILP) formulation for the LSP allocation problem is outlined and two different objective functions are proposed and evaluated. The solutions from these models are then used to determine the maximum link utilisation in the network. Secondly, the MILP models for MPLS are verified using the well known ns-2 simulator [50] and the statistics for packet loss in the network are presented. Thirdly,

an LP model from the previous chapter is formulated and is used to minimise the number of explicitly routed LSPs. The experimental results for large networks are presented.

The rest of the chapter is organised as follows: Section 4.2 gives a brief overview of MPLS technology. Section 4.3 outlines two different formulations for the optimisation problem with various different objective measures. Section 4.4 describes the setup for carrying out the experimental work. Section 4.5 describes the results obtained from the calculation and the simulation and also provides some detailed discussion of the results. Finally, section 4.7 summarises the chapter.

4.2 MPLS Overview

4.2.1 MPLS Framework

MPLS was initially developed to solve bottleneck problems caused by searching IP forwarding tables. Native IP routers perform longest-prefix matching for the destination IP address of every IP packet. As the number of entries in the forwarding table increases, the execution time of the longest-prefix matching search takes more and more time to obtain a result. This searching process has proven to be a bottleneck in router speed. Hence, the concept of MPLS was introduced to improve IP packet transmission performance.

In an MPLS enabled domain, incoming IP packets are assigned a short fixed-length "label" by a Label Edge Router (LER). An LER is usually located at the boundary of an MPLS enabled domain. Packets are forwarded along a "label switched path (LSP)", where each MPLS router, commonly known as a Label Switch Router (LSR), makes forwarding decisions based on the contents

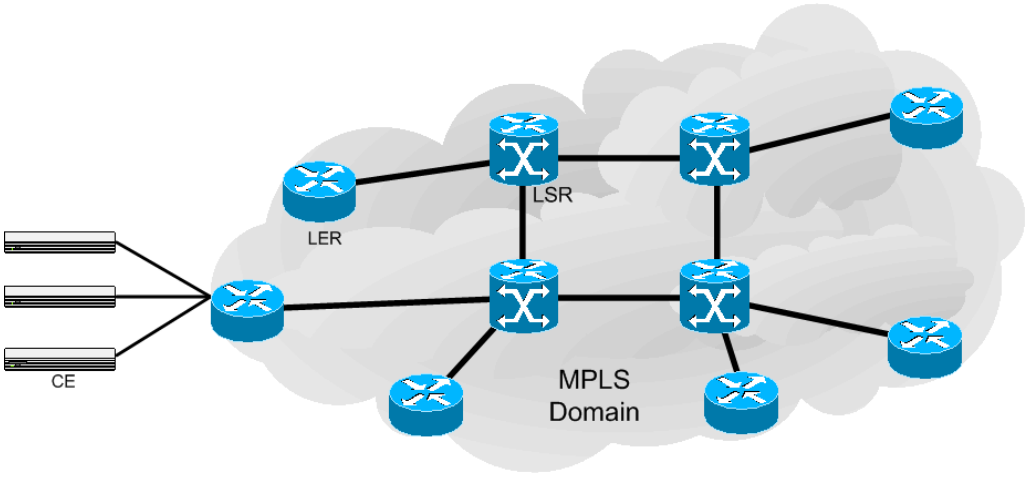

Figure 4.1: Typical MPLS Network Architecture

of the label. A flat MPLS topology is given in Figure 4.1.

Label Switched Paths (LSPs) are established by network operators for a variety of purposes, such as to guarantee a certain level of performance, to route around network congestion, or to create IP tunnels for network-based virtual private networks. In many ways, LSPs are no different than circuit-switched paths in ATM or Frame Relay networks, except that they are not dependent on a particular Layer 2 technology.

An LSP can be established that crosses multiple Layer 2 transports such as ATM, Frame Relay or Ethernet. Thus, one of the true promises of MPLS is the ability to create end-to-end circuits, with specific performance characteristics, across any type of transport medium, eliminating the need for overlay networks or Layer 2 only control mechanisms.

The initial goal of label based switching was to bring the speed of Layer 2 switching to Layer 3. Label based switching methods allow routers to make forwarding decisions based on the contents of a simple label, rather than by performing a complex route lookup based on a destination IP address. This initial justification for technologies such as MPLS is no longer perceived as

the main benefit, since Layer 3 switches (ASIC-based routers) are able to perform route lookups at sufficient speeds [61] to support most interface types. Nevertheless, MPLS still brings many other benefits to IP-based networks, they include:

- Traffic Engineering - a capability to select a path which satisfies the traffic requirements and to set performance characteristics of a class of traffic.
- VPNs - a capability to create IP tunnels throughout the managed network at Layer 2.
- No multiple layers to manage. Classical overlay models where SONET/SDH, ATM and IP are used by carriers and are required to be individually managed. MPLS allows the service carriers to use MPLS, carriers can migrate many of the functions of the SONET/SDH and ATM control plane to Layer 3, thereby simplifying network management and network complexity. Eventually, carrier networks may be able to migrate away from SONET/SDH and ATM all-together, increasing network efficiency due to the elimination of ATM headers.

With MPLS technology, network operators have the freedom to control how traffic is being mapped onto network resources. Network operators have capabilities to assign traffic forwarding behaviour inside an MPLS domain based on the type of traffic or SLA agreements with their customers. Once traffic is classified at the edge of the network, packets belonging to the same forwarding behaviour will be tagged with a label such that they follow the same LSP forwarding path and this classification will not be changed as the packet progresses through the network.

4.2.2 Achieving Traffic Engineering in MPLS Networks

MPLS Traffic Engineering is concerned with the selection of suitable LSP paths based on traffic requirements. One of the simplest requirements is the bandwidth requirement. This requirement states that the links included in the LSP must be able to provide a certain amount of bandwidth for this LSP.

Traffic Engineering in MPLS networks can be classified in several different ways. One of the most prominent is the classification based on time scale. With *off-line* calculation, MPLS paths are computed during network setup or periodically, based on a traffic matrix *a priori*. The *online* calculation involves of computing MPLS paths as the requests arrived [62].

Off-line calculation is performed over a longer time scale, such as weeks or months [62], and it usually requires knowledge of the network topology and an associated traffic matrix. After this information is obtained, the calculations can be carried out. As a result, a set of static paths, LSP paths, are set up manually by the network administrator to perform the routing in the network. The initial work of path and capacity allocation in communication networks are given in [14] and [13]. In [44], Garcia et al propose the use of a 3-stage optimisation to allow precise optimisation of bandwidth utilisation and to provide backup routes in the case of link failure.

Included in the on-line calculation is the implementation of a widest shortest path algorithm to find a suitable LSP path. Upon receiving a request, a calculation to find a path is invoked, based on the traffic requirements (bandwidth, delay or jitter). Once a suitable path has been identified by a Label Edge Router (LER), signalling protocols such as LDP or RSVP can be used to set up the path.

An LSP can be statically configured (Explicit Route LSP) or dynamically

built. An explicit route LSP is configured by the network administrator (mainly based on solutions obtained by off-line calculations). A dynamically built LSP utilises signalling mechanisms such as the Label Distribution Protocol (LDP) or Resource Reservation Protocol (RSVP) via Constrained Shortest Path Forwarding (CSPF). In [40], Kodialam et al presented a new algorithm for dynamic routing with bandwidth guaranteed tunnels where the requests arrive one-by-one and there is no *a priori* knowledge regarding future requests. The solution for the new tunnel should not "interfere too much" with a route that may be critical to satisfy a future demand. The request rejection rate is used as a performance metric. Another approach that adaptively balances the load among multiple paths based on measurement and analysis of path congestion called MPLS Adaptive Traffic Engineering (MATE) was proposed in [39]. A comparison of some of the MPLS dynamic routing algorithms found in the literature is given in [63]. Similar works that aim to minimise the interference to existing flows and critical routes are given in [64], [65] and [41].

4.3 Proposed MPLS Optimisation Models

This section introduces two MPLS routing allocation problem formulations suitable for an off-line calculation, namely a single path multi-commodity flow problem (MinCost) in section 4.3.1 and the maximum residual capacity single-path problem (MaxResidual) in section 4.3.2. The performance measure in this case can be the network utilisation, total residual bandwidth and the packet loss.

The classical multi-commodity flow problem (MCF) problem as described in [10] or [56] addresses the problem of how to send a specified amount of commodity from a source to a destination such that the total cost is minimised.

In this problem, a graph, which is a collection of nodes and links, is specified and the links have an associated cost. The amount of the commodity can be associated with the size of the traffic flow in the routing problem. The link cost can be associated with any form of routing metric or it may relate directly to real dollar costs. The classical MCF formulation can be represented by a link-flow formulation or by a path-flow formulation. In this thesis, the path-flow formulation is used for easier understanding and representation - it also has certain advantages over the link-flow formulation which have been considered in other contexts.

Although the traditional MCF path-flow solution can be used for network resource allocation, the formulation typically requires some adjustment via additional constraints. In particular, the MCF solution does not consider that the flow for an OD-Pair should be limited to a single path. The general formulation allows an OD-Pair flow to be split into two or more separate routes to fill the cheaper route first. In the context of this chapter, an OD-Pair needs *to be routed on one and only one path from the source to the destination*. Hence, additional constraints are required to prevent splitting of the OD-Pair flows.

The network is modelled as a uni-directional graph. A path is defined as a series of links that connects the origin and the destination node of the associated OD-Pair. In our analysis, these paths are calculated for *each* OD-Pair using the *k*-shortest path algorithm due to Yen [66]. The following notation is used in the formulation:

- $\delta_i^k(P)$ is equal to 1 if link i is contained in path P of OD-Pair k and 0 otherwise.
- d^k is the traffic demand for OD-Pair k.
- u_i is the capacity of link i.

- $f(P)$ is a decision variable which denotes the amount of flow on path P.
- $c(P)$ is the sum of link metrics or link costs along path j.
- $a^k(P)$ is a binary decision variable which takes the value 1 if path P is used to carry flow for OD-Pair k and 0 otherwise.

4.3.1 Single-Path Multi-Commodity Flow Problem (Min-Cost)

$$\text{Minimise} \sum_{k \in K} \sum_{P \in P^k} c(P) f(P) \tag{4.1}$$

subject to

$$\sum_{k \in K} \sum_{P \in P^k} \delta_i^k(P) f(P) \leq u_i \qquad \text{for all } i \in E \tag{4.2}$$

$$\sum_{P \in P^k} a^k(P) f(P) = d^k \qquad \text{for all } k \in K \tag{4.3}$$

$$\sum_{P \in P^k} a^k(P) = 1 \qquad \text{for all } k \in K \tag{4.4}$$

$$a^k(P) \in (0,1) \quad f(P) \geq 0 \qquad \text{for all } P \in P^k \text{ for all } k \in K$$

The above formulation is the single path multi-commodity flow problem version. Whilst the objective is to *minimise* the total cost to transfer the commodity from the source node to the destination node equation (4.1), the total amount of commodity flowing on a particular link cannot exceed the link capacity equation (4.2). These constraints are also known as the *bundle con-*

straints. Equation (4.3) specifies that the total flow carried on all available paths for a particular OD-Pair must be equal to the traffic demand for that pair (no loss is permitted). Equation (4.4) enforces the requirement that only one path is permitted to be used to carry the demand from the source to the destination. This formulation is also presented in [60].

However, multiplying two unknown variables in equation (4.3) results in non-linear constraints. The constraints need to be written differently to form linear constraints. Since only one path can carry the whole OD-Pair flow, there exists one and only one of the $f(P)$s which must be non-zero. Furthermore, the value of this non-zero $f(P)$ must be equal to d^k. Hence, equation (4.3) can be re-written as follows:

$$\sum_{P \in P^k} d^k a^k(P) = d^k \qquad \text{for all } k \in K$$

Dividing both sides by d^k yields the same constraint as equation (4.4). Hence, equations (4.3) can be discarded because they are redundant. Substituting $f(P) = d^k a^k(P)$ in the above problem for simplicity gives the reduced formulation as follows:

$$\text{Minimise} \sum_{k \in K} \sum_{P \in P^k} c(P) d^k a^k(P)$$

subject to

$$\sum_{k \in K} \sum_{P \in P^k} \delta_i^k(P) d^k a^k(P) \leq u_i \qquad \text{for all } i \in E$$

$$\sum_{P \in P^k} a^k(P) = 1 \qquad \text{for all } k \in K$$

$$a^k(P) \in (0,1) \qquad \text{for all } P \in P^k \text{ for all } k \in K$$

The lower bound for this model can be determined by eliminating single path constraints. The model then becomes an original multi-commodity flow problem that can readily be solved using the simplex method.

4.3.2 Maximum Residual Single-Path Problem (MaxResidual)

To minimise the delay and packet loss while sending packets from the source to the destination, it is necessary that the traffic be spread such that none of the links is congested. In this formulation, the objective is to avoid a bottleneck in the network. In other words, the objective is to *maximise* the minimum residual link capacity. The residual link capacity of link i, R_i, is defined as the difference between the link capacity and the total traffic carried on that link. A common R is defined as $\min_i R_i$ for every link.

$$\text{Maximise } R \tag{4.5}$$

subject to

$$\sum_{k \in K} \sum_{P \in P^k} \delta_i^k(P) d^k a^k(P) + R \leq u_i \qquad \text{for all } i \in E \tag{4.6}$$

$$\sum_{P \in P^k} a^k(P) = 1 \qquad \text{for all } k \in K \tag{4.7}$$

$$R \geq 0 \qquad a^k(P) \in (0,1) \qquad \text{for all } P \in P^k \text{ for all } k \in K$$

In the initial feasible solution, some links might have no residual capacity

at all ($R_i = 0$ for some links, hence $R = 0$). In order to perform practical calculations of the required variables in this project, a well-known standard package for solving Linear Programming problems, known as CPLEX [67] has been used. Through the branch and bound process in CPLEX, a better solution can be obtained by moving flows away from these links to "push up" the values of R_i (equation (4.5)) resulting in a better objective value. Equation (4.6) denotes the total flows on the link and the minimum residual capacity must be less than the link capacity. The single path flow allocation is enforced by introducing equation (4.7).

4.3.3 The Model's Application in MPLS

When the above formulations are solved using an LP solver (such as CPLEX [67]), the outputs will be a series of decision variables $a^k(P)$. OD-Pair k will be routed on path P, whose value of $a^k(P)$ is equal to one. In MPLS, this can be easily implemented by setting up an explicit route. MPLS enables the network administrator to define the complete set of intermediate nodes along the LSP path.

Upon receiving a request for an explicit route in MPLS, the Label Edge Router (LER) will send out a message to its neighbour to pin down the path to be used to reach the desired destination. This neighbour then propagates the request down the path to the destination. When the request is successful, the LER creates a label associated with this path to the destination.

4.4 Experimental Setup

Figure 4.2 depicts the network topology used in this study. It consists of 8 routers. Each of these routers is connected to a single workstation that acts as a traffic generator and a traffic sink. 10 Mbps and 100 Mbps links are used to connect between the routers. 300 Mbps links are used to connect between the workstation and the router. 10 Mbps and 100 Mbps links are assigned OSPF weights of 1 and 10, respectively. Assuming a traffic demand exists between every pair of workstations, there will be a total of 56 possible traffic flows. The size of these traffic flows has been generated randomly for this experiment. This topology is chosen because it is simple for the simulation purposes and yet it provides sufficient alternate paths for MPLS routing.

Throughout this study, 3887 different traffic matrices have been used; each will be referred to as an *instance*. These instances are then categorised into 41 different groups according to their load level. These groups will be referred to as group 10 to group 50. Figure 4.3 depicts the number of instances that belong to the same group. Whilst most of the groups have more than 80 associated instances, there are a few groups that only have a few instances associated with them. This difference will affect the range of the confidence intervals obtained – as discussed in section 4.5.

Figure 4.3 also shows the average total demand for each individual group. An increasing total demand from 26.5 Mbps (group 10) to 120 Mbps (group 50) is intended to simulate the network under different load conditions, from lightly loaded up to a saturated condition. Instances that belong to groups 10 to 25 are considered to be light loads. Those in groups 26 to 40 and groups 41 to 50 are considered as moderate and heavy loads, respectively.

In this study, OSPF will be used to provide benchmark performance, be-

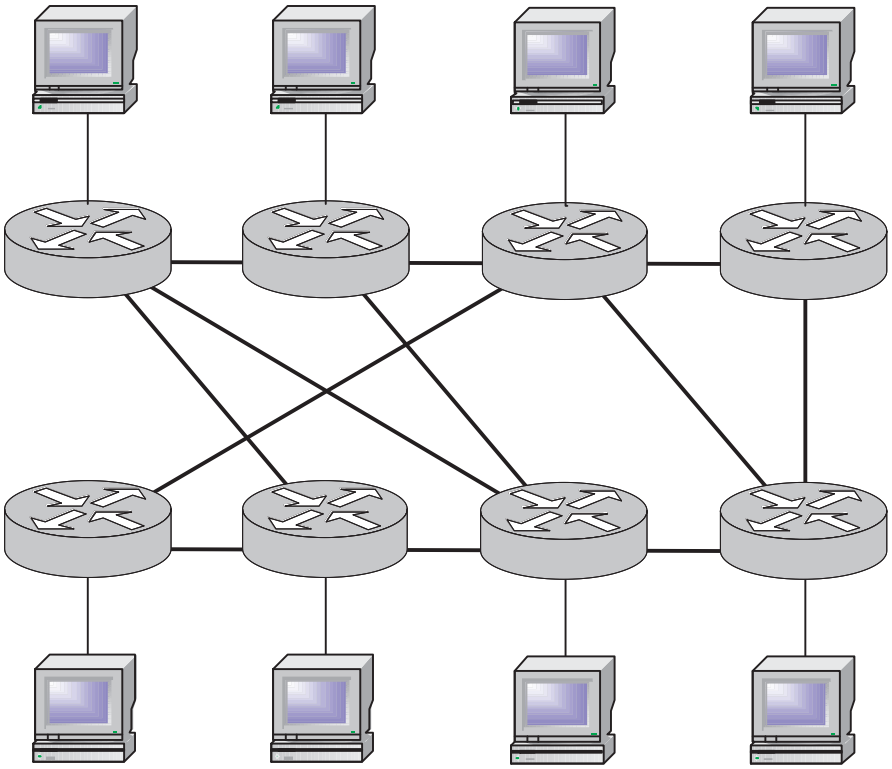

Figure 4.2: Simulated topology

cause it is widely used in practice. Nowadays, OSPF link metrics are often used for establishing LSPs in MPLS; the MPLS Traffic Engineering feature is rarely utilised. In addition, the performance of the two MILP schemes described earlier will be compared. The link utilisation in the OSPF case is calculated by assuming an even splitting whenever equal cost paths to the destination exist. This is commonly achieved by employing Equal Cost Multi Path (ECMP) in a router's forwarding plane to balance the load.

ns-2 with the MPLS extension module [50] was used as the simulator in this study. Since such routing is only done in core networks (i.e. among the 8 routers, see Figure 4.2), only these 8 routers need to be included in the MPLS domain. Explicit routes exist within these 8 nodes in the MPLS domain.

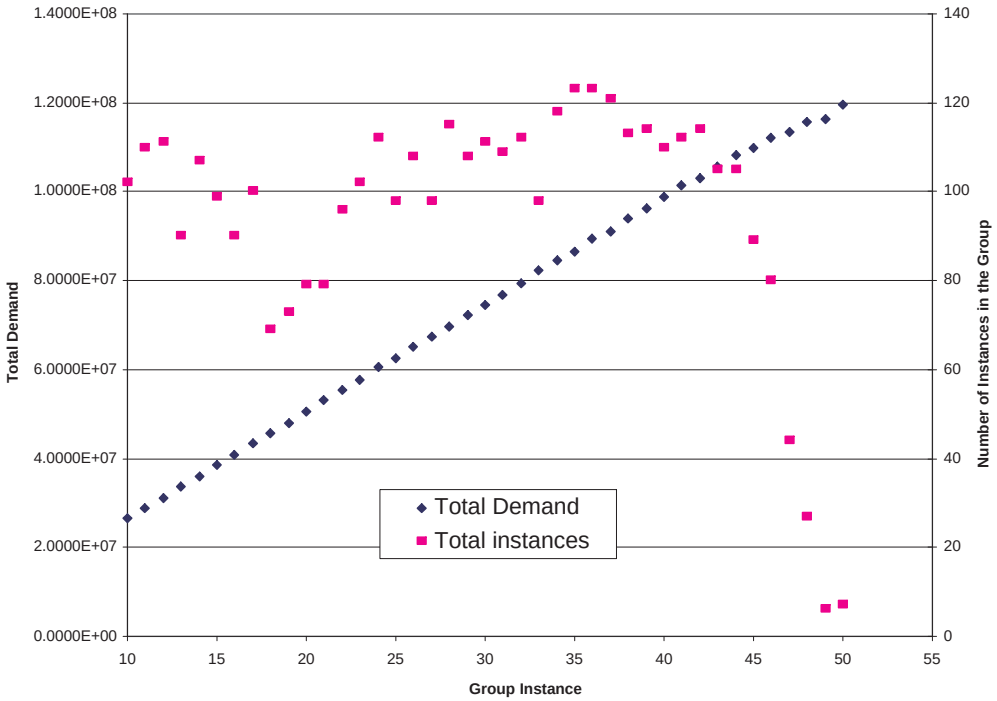

Figure 4.3: Grouping profiles

OD-Pairs are modelled as constant bit rate (CBR) sources. This has been done for simplicity reasons. The CBR rate is governed by the size of the individual traffic flows and the packet size. The packet size is chosen to be 500 bytes. The packet inter-arrival time is uniformly randomised to avoid synchronisation among traffic sources. The CBR sources are activated at the start of the simulation and CBR datagrams are carried by UDP packets. On the receiving end, a loss monitor is set up to collect statistics on the number of packets that are received or lost.

The ns-2 simulation is run for 20 seconds. The first one second is regarded as the simulation warm-up period and measurements during this period are discarded. At the end of the simulation, the statistics for packet lost and received packets for each individual OD-Pair are collected and analysed.

4.5 Results and Discussion

This section presents the calculation results after evaluating the MILP models using the simulator. The results that have been obtained are discussed in Section 4.5.1. The model solutions are also used as the inputs for the ns-2 simulation. The simulation results are discussed in Section 4.5.2.

4.5.1 Models Comparison

The solution of the MILP formulation, which is a set of paths used to carry flows for each of the OD-Pairs, is used to determine the amount of traffic on every link in the network. Given that the link capacities are known, the utilisation can be calculated directly. The parameter of interest in this comparison is the maximum link utilisation. It is used to determine the severity of a congestion bottleneck in the network.

Figure 4.4 depicts the maximum link utilisation when three different routing schemes are used with the different load groups. The 95% confidence intervals for the results are shown as error bars in the graphs. In the lightly loaded region, the OSPF performance and the MPLS MinCost are exactly the same. This can be explained as follows: OSPF is a routing protocol that routes the traffic without considering link capacities, instead it uses the “distance” metric known as an OSPF weight on the links. A path to a given destination that has the lowest sum of OSPF weights will be chosen. The MPLS MinCost model will do exactly the same; it will route the traffic on the path with the lowest cost. The bundle constraints in this particular situation have not “kicked in” yet because none of the links is fully utilised at this point. As a result, OSPF and MPLS MinCost routing both yield maximum link utilisations varying from 40% to 95%. As expected, the MPLS MaxResidual has the

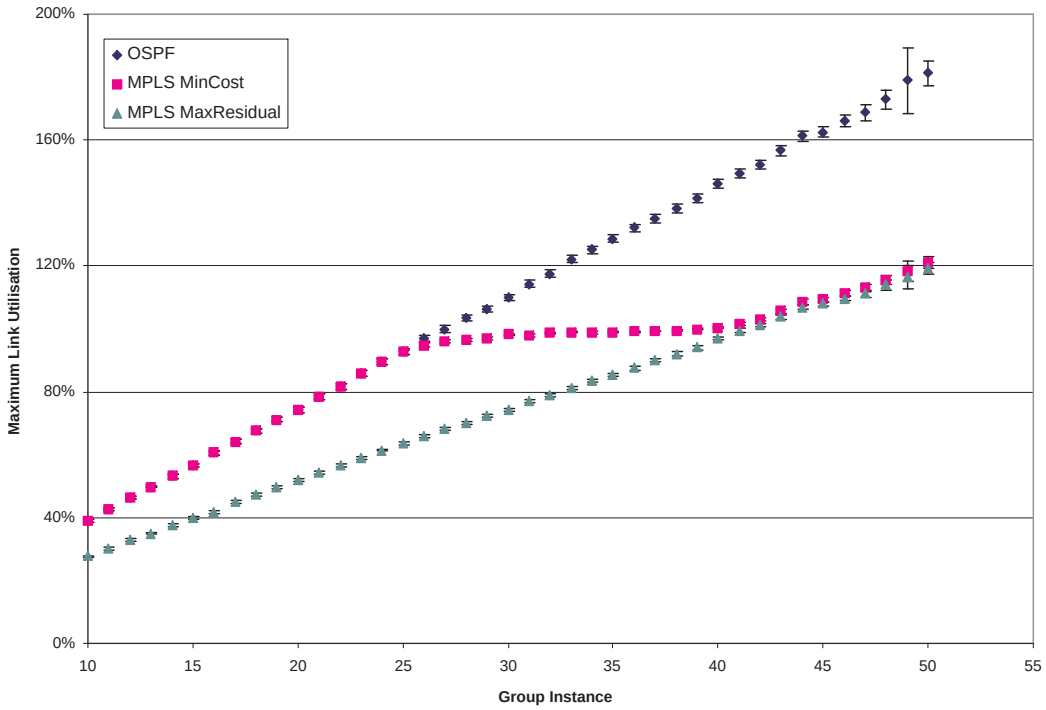

Figure 4.4: Maximum Link Utilisation in the 8 node test network

lowest utilisation as it spreads the load throughout the network (30% to 65% utilisation).

Figure 4.4 also shows the results when the load is increased. With the increased load, OSPF routing yields a proportional increase in the maximum link utilisation (correlate this with Figure 4.3). In cases where the network is moderately loaded, the bundle constraints in MPLS MinCost formulation start to take effect. They restrict the amount of traffic on a link to be less than the link's capacity. The overflow traffic will be routed on different paths with available capacity. The maximum link utilisation is "kept" at 100% for MPLS MinCost. The flat line in the MPLS MinCost curve in Figure 4.4 shows that the maximum link utilisation does not change even when the load is increased to 40% (moderate network loads). The MPLS MaxResidual still

performs better than OSPF and MPLS MinCost.

However, as the load increases further, the MPLS MaxResidual advantage slowly diminishes. Its maximum link utilisation approaches that of MPLS MinCost. In the highly loaded region, the network is completely saturated. There is no more capacity to accommodate the additional traffic. To deal with the infeasibility problem in MPLS formulations, virtual capacities are made by scaling up the real capacity by a specified factor. In this case, both MPLS models give a similar performance (see Figure 4.4).

For this problem size, the calculation time for the MPLS MinCost formulation is less than one second using CPLEX. This processing time can be considered negligible. Solving the MPLS MaxResidual formulation takes longer. The optimal solution to within 20% of optimality can be obtained in less than 5 seconds, on average, by using CPLEX. The optimality is calculated assuming no integrality constraints. In some of the heavily loaded cases, the solution search has to be terminated after 40 seconds has elapsed, since it is rare that any improvements can be made after 40 seconds. Surprisingly, the calculated link utilisation still falls into the trend line given by the sub-optimal solutions.

The difference in the run-time is attributed to the nature of the problems. The decision variables in the MPLS MinCost formulation are all binary values. This yields a pure integer programming problem. However, the MPLS MaxResidual formulation has an additional decision variable, R, which takes a continuous value. Hence, the problem becomes a mixed integer linear programming problem.

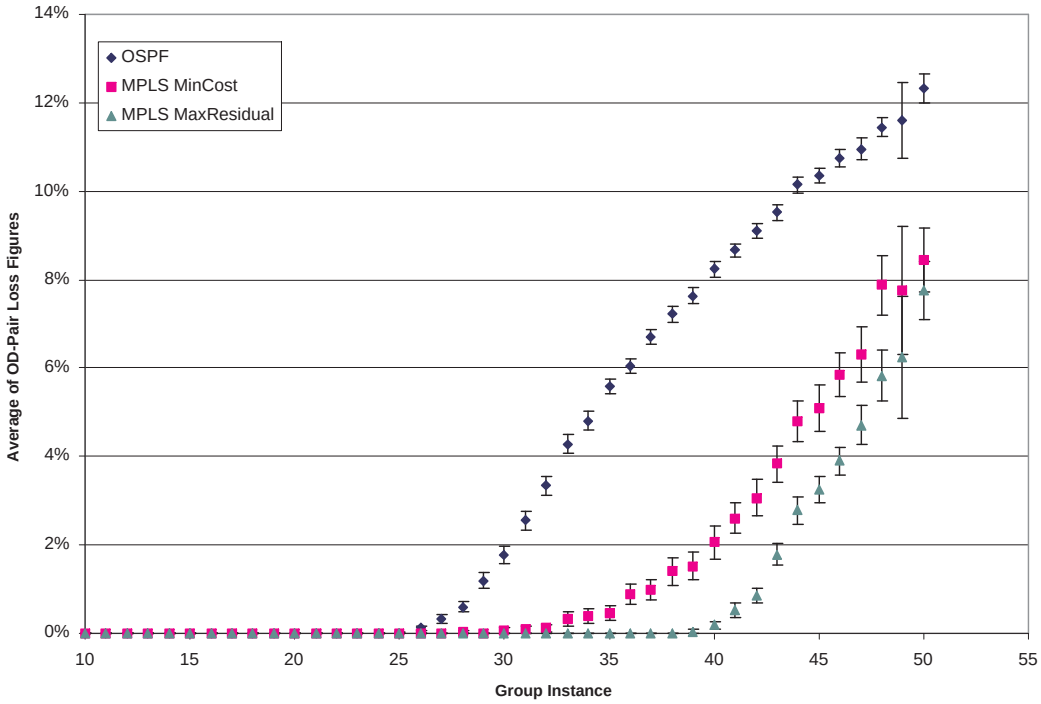

Figure 4.5: Average loss probability in 8 node test network

4.5.2 Simulation Results

A simulation is carried out to show the performance improvement of MPLS explicit routing based on the MILP models in comparison to OSPF routing. The parameter of interest is the number of packets that are lost. For *every* instance, the number of packets lost per OD-Pair is monitored. The average of these figures is then grouped, averaged and plotted. Another measure is the maximum OD-Pair loss probability, which is obtained by taking maximum OD-Pair loss probability from a simulation instance. The averages of these maxima are then grouped, averaged and plotted. These results are summarised in Figure 4.5 and 4.6. The 95% confidence intervals are given as error bars in these graphs.

Figure 4.5 shows that OSPF routing starts to exhibit packet loss when

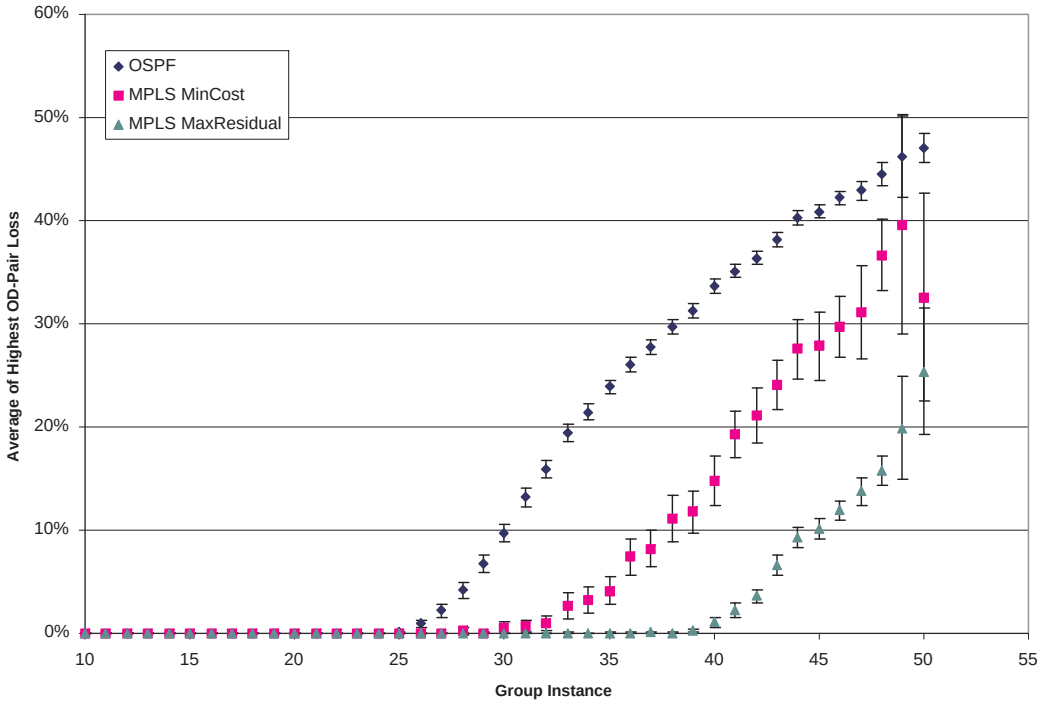

Figure 4.6: Maximum loss probability in 8 node test network

the network is moderately loaded (group 25). MPLS MinCost model slightly extends the non-loss region up to group 31. However, the most noticeable improvement is shown by MPLS MaxResidual having no loss until group 39. With MPLS MaxResidual, the network can support about 45% more traffic without experiencing packet loss compared with OSPF. When the network is heavily loaded up to its saturation point, the advantages of the MILP models slowly diminish (both MPLS curves start to approach the OSPF curve).

In terms of the maximum packet loss statistics as shown in Figure 4.6, in group 39, MPLS MaxResidual still can perform without experiencing packet loss, whilst OSPF and MPLS MinCost exhibit 30% and 14% loss, respectively. Although the average loss for MPLS MaxResidual is just slightly better than MPLS MinCost, the MPLS MaxResidual gives much better results in terms

of the maximum OD-Pair loss probability. In MPLS MaxResidual, most of the OD-Pairs have similar losses because most of the links are saturated. None of them experience more loss than the others. The 95% confidence intervals in MPLS MaxResidual are tighter than those of MPLS MinCost.

It can be concluded that when the network is moderately loaded, having a model that attempts to free up bottlenecks in the network can be very beneficial. This is verified in terms of the average OD loss and maximum OD loss results. MPLS MaxResidual performs substantially better compared to MPLS MinCost and OSPF. It is advisable to "allocate" residual capacity to protect against congestion due to traffic measurement uncertainty and bursty traffic. However, when the network is heavily loaded, the experiment shows that the advantage of MPLS MaxResidual slowly diminishes. Together with an increasing computation time when solving the MPLS MaxResidual formulation, MPLS MinCost may outweigh MPLS MaxResidual.

In the heavily loaded region, the loss probabilities do not exactly fall into the trend. A small number of instances in these groups contributes to the deviation. A smaller number of instances also results in wider confidence intervals. Most of the groups have 80 instances each, however the last few groups only have less than 30. Maximum time restrictions for solving the MPLS MaxResidual formulation, which yield sub-optimal solutions, may also contribute to the deviation.

Having the available capacity to reroute traffic is particularly important in all traffic engineering methods. In a situation, where there is no capacity available to reroute the traffic, a different congestion management scheme needs to be employed. Congestion control mechanisms that automatically adjust the sending rate depending on the network load is required.

4.5.3 Optimisation of an Operational Network

It is important during subsequent optimisations that only a few LSPs need to be re-configured and re-established. A complete LSP re-shuffling is generally not acceptable, because of the introduced disturbance to customers' flows. The MPLS optimisation process can be influenced to take this factor into consideration.

The MILP formulations in this thesis can be extended as follows: The first alternative is to restrict the elements in the path list. The path list for the OD-Pair, whose LSPs are not permitted to be re-configured, should contain only one path. This path must correspond to the path that is currently taken by the LSP. A second alternative approach is to pre-set the decision variable $a^k(P)$ to unity, ensuring that OD-Pair k uses path P. Path P corresponds to the path that should not be changed.

4.6 Minimising the Number of Explicitly Routed LSPs

This section outlines a technique that can be used to reduce the number of LSPs that need to be explicitly routed.

4.6.1 Relying on IGP metrics to compute LSPs

Given that configuring a large number of LSPs can be a daunting task for a network administrator, it is preferred that the LSPs are built based on IGP metrics. In [46], it is shown that it is possible to route most of the traffic along optimised shortest paths, while MPLS is only partly introduced to complement the optimisation process. In [47], Mulyana et al introduce a hybrid

IGP/MPLS traffic engineering method based on genetic algorithms, which can be considered as an off-line TE approach to handle long or medium-term traffic variations in the range of days, weeks or months. In their approach, the maximum number of hops an LSP may take and the number of LSPs which are applied solely to improve routing performance, are treated as constraints due to delay considerations and the complexity of management. Hence, in MPLS networks, it is highly desirable to route the traffic (build the LSPs) based on the IGP mechanism (link based method). Only when optimality is needed, then MPLS tunnels should be deployed.

To recap from the previous chapter, a splitting ratio, R_{nl}^{nt}, at node n defines the amount of the outgoing traffic proportion directed to a particular destination t that needs to be sent on the router's interfaces to its neighbour l. The splitting ratio is calculated based on the primal solution of the problem (see chapter 3).

Our method approximates the ratio splits by deciding whether to replace it with a single path flow ("1-0" case). The majority of traffic (when there is no splitting ratio) is routed based on LSPs that are established based on the IGP metric. Explicitly routed LSPs (when there is no splitting ratio) are used for those flows on paths that have multiple equal length paths (based on an IGP metric) to the destination. Thus the number of explicitly routed LSPs equals the number of "1-0" splitting ratios. This method has similarities to those networks where MPLS is reserved for traffic that has a particular status separate from the bulk of best effort traffic, such as VPN traffic or higher QoS class traffic.

4.6.2 Experimental Results

We tested our method on networks made up from "standard" AT&T networks merged together. The first network has 56 nodes and 200 unidirectional links. The second has 90 nodes and 314 unidirectional links. Thus, assuming a traffic flow between each node, there are 3080 flows for a 56 node network and 8010 flows for a 90 node network. The traffic matrix was generated randomly using a uniform distribution between 0.1 and 7.9, the total throughput being just large enough to heavily overload the network but still feasible. We also used the Zegura 2-level hierarchical network, which consists of 100 nodes and 374 unidirectional links (9900 flows)[68].

To make the hybrid solution as close to optimality as possible, we must minimise the number of splitting ratios. This has the added benefit of reducing the number of LSPs. To do this one must "seed" the LP solver [67] with a good set of starting weights. This requires only that we ensure that the starting solutions are mostly single paths for each flow. The reasons for this are somewhat subtle and are, in fact, explained in an accompanying paper [21]. To ensure that each flow has a unique shortest path is actually trivial. In the tables below, when the initial link costs are set (in the solver - NOT in the real network) to the inverse of their capacity (InvCapOSPF) or randomly (RandOSPF), it is easy to check that all flows are using their unique shortest path. The uniform distributions for capacities are listed in Table 4.1. In this case the final number of LSPs after optimisation is small. In the case of UnitOSPF the number of initial equal cost paths is 999.

Another important point is that the number of LSPs should remain small in subsequent wholesale re-optimisations, i.e. where the traffic matrix is completely changed. This required that we used the final basis of one optimisation

Capacities	Cost Type	Shortest path Max Util	Max Util after optimisation	Number of LSPs
S	InvCapOSPF	243.6%	102.5%	93
M	InvCapOSPF	266.3%	106.1%	157
D	InvCapOSPF	204.3%	101.5%	88
S	UnitOSPF	229.0%	110.7%	813
M	UnitOSPF	331.3%	104.6%	748
D	UnitOSPF	372.5%	118.1%	823
S	RandOSPF	270.5%	105.2%	176
M	RandOSPF	361.2%	109.0%	84
D	RandOSPF	552.9%	108.4%	74

Table 4.1: Maximum utilisation reduction on the 56 node network
Legend: S - 400-600 units M - 200-800 units D - 100-900 units

Capacities	Cost Type	Shortest path Max Util	Max Util after optimisation	Number of LSPs
S	InvCapOSPF	355.4%	105.5%	194
S	UnitOSPF	262.1%	135.9%	1483
S	RandOSPF	295.7%	108.5%	174

Table 4.2: Maximum utilisation reduction on the 90 node network
Legend: S - 400-600 units

process as the advanced basis for the next one. Changing the demand matrix also required small amounts of demand re-scaling (0.95 - 1.05) such that the problem was LP feasible. In each case, we found that the number of dual prices, split paths and split ratios was approximately the same.

We now further investigate a case in which the link capacities are all the

Capacities	Cost Type	Max Util after optimisation	Number of LSPs
E	InvCapOSPF / UnitOSPF	107.2%	820
E	RandOSPF	106.9%	160

Table 4.3: Performance under equal link capacities (E)

same (see Table 4.3). To get rid of all the multiple paths caused by InvCapOSPF or UnitOSPF, (they have the same effect for this case) we propose that we randomise the initial weights as before. We see that the level of optimality is practically identical but the number of LSPs is now only 160.

4.7 Conclusions

This chapter has outlined two MILP models for the LSP allocation problem in an MPLS domain. These models make use of the explicit routing capability of MPLS. The output of these models has been compared against default OSPF routing. The calculation and simulation results for many test scenarios has been presented to verify the formulations. It is beneficial to maximise the minimum residual capacity in the network in order to spread the load which, in turn, balances the link utilisations and reduces the packet loss. The MPLS MaxResidual outperforms OSPF and MPLS MinCost when the network is lightly and moderately loaded. With high loads, it loses its advantage; MPLS MinCost may be a more attractive solution given its faster calculation time. These two MILP models can be extended to allow the optimisation of operational networks without having to reconfigure many existing LSPs. Furthermore, we have also shown that it is possible to reroute a small number of explicitly routed MPLS when the network operating point is changed by a small margin.

Chapter 5

Traffic Engineering in Multi-class Traffic Networks

The pure IGP weight-setting approach is limited in that it can only produce one path or set of equal cost paths for each origin-destination pair and so it is not possible to independently route different service classes. Yet a scheme that allows real-time or high priority traffic to be routed via the lowest delay path and best-effort traffic via the highest bandwidth path, whether or not the two are coincident, is very attractive.

This chapter presents an optimisation method to traffic engineer networks carrying different classes of traffic. The Mixed Integer Programming model is based on the classical multi-commodity flow problem. Although some initial input data (such as the topology and a traffic matrix) is required, the method can be directly applied and no protocol or router modifications are necessary. The method reconfigures the routing pattern in a way that allows the network to carry more traffic and capacity expansion can be delayed. It is shown here that the method brings up to a 50% improvement in the maximum link utilisation when it is compared to the Inverse Capacity metric, which represents

the typical default Cisco routing metric. Furthermore, in the event where high priority traffic "over-subscribes", above its limit and causes the best effort traffic to be congested, we show that by just modifying a few weights we can effectively redistribute the best effort traffic.

5.1 Introduction

Whilst Internet content is still largely delivered using the best effort service, there are emerging applications requiring different services and these services may require bandwidth, delay or jitter guarantees. For example, mission critical applications require close to 100% packet delivery rates and cannot tolerate delay and jitter more than a certain threshold. Real-time applications such as VoIP or video streaming applications require minimum delay and jitter while they are more resilient to loss (non-elastic traffic). On the other hand, non-time critical applications, such as email, can tolerate delay and scarce bandwidth. We can classify mission critical and real-time traffic as high priority traffic and non-critical traffic as Best Effort (BE or elastic traffic) traffic.

More formal definitions of traffic classification are given as a part of DiffServ specifications in the relevant RFC, [69]. Diffserv is a specification that takes the IP TOS (Type of Service) field, renames it the DiffServ Code Point (DSCP) byte, and uses it to carry information about IP packet service requirements. DSCP is used by each router along its path to give it a particular forwarding treatment. The main objective of DiffServ work is to classify traffic based on the DSCP standards and assign specific forwarding treatment to be received by best-effort traffic. DiffServ mechanisms allow network providers to allocate different levels of service to different users of the Internet in a

scalable manner.

MPLS, as discussed in the previous chapter, is a switching technology. MPLS specifies ways that Layer 3 traffic can be mapped to connection-oriented Layer 2 transports like ATM and Frame Relay. MPLS adds a label containing specific routing information to each IP packet and allows routers to assign explicit paths to various classes of traffic. It was initially developed to improve efficiency in IP routing. However, it is not true anymore, as IP routing can be done as quickly as MPLS switching, thanks to ASIC developments. However, MPLS is now used for traffic engineering and VPNs [70] [71].

What is MPLS in relation to DiffServ? They are two separate standards that help achieve Quality of Service in the Internet. DiffServ requires mechanisms such as traffic measurement, traffic shapers, packet markers to be used at the boundaries of the network. It then aggregates traffic flows into aggregate flows based on the required forwarding behaviour. MPLS, with its label-switching capability, then assigns and switches packets through a specific path i.e. a Label Switched Path (LSP), which has been previously established.

The work in this chapter focusses on traffic engineering a network that carries multiple classes of traffic. It is noted that traffic engineering issues have become more complicated due to the requirement of the network to support multiple classes of traffic, so that service quality is a requirement that must be satisfied. The issue of how DiffServ actually classifies the traffic flows and forwarding treatment is not in the scope of this chapter. Here, we assume that there are only two types of traffic, high priority traffic and best effort traffic; both have been classified and only require suitable LSPs to route them to the intended destination. High priority traffic requires a minimum outgoing rate to be specified as Expedited Forwarding (EF) traffic. A virtual leased line, in

which the ISP has to conform to a Service Level Agreement (SLA), can also be categorised as high priority traffic. It has been shown that this approach can be an effective technique to support traffic with low jitter requirements [72]. These higher priority LSPs may be routed arbitrarily. This model corresponds to either an IntServ/MPLS network [73] with reserved flows mapped into higher priority LSPs or a DiffServ network where the high priority LSPs carry the higher classes of traffic (either or both EF and AF) [70] [71].

At any time, the network has to be able to carry the EF traffic which must be fully restored and BE traffic which should be restored as much as possible above a certain percentage level in the event of a link failure or when an optimisation takes place on an operational network. Given the number of networks migrating to MPLS technology, we utilise the explicit routing feature in MPLS to route EF traffic. Best effort traffic can be routed using a native intra-domain protocol such as OSPF, which works based on the shortest path paradigm as explained in previous chapters. Alternatively, it can also be routed on LSPs that are built based on an IGP metric. Regardless, both approaches require a set of shortest path metrics.

The first contribution of this chapter is a two-phase Mixed Integer Linear Programming (MILP) model formulation for optimising multi-class network traffic. The first phase of the model is based on the classic multi-commodity flow problem. It is used to determine a suitable path for every traffic class and every origin and destination pair (OD-Pair). A suitable path is a path that is able to carry traffic at levels that are higher than its restoration percentage. The problem formulation allows a network provider to specify possible paths to use or to use a path generation algorithm for each individual OD-Pair in the network. Additional constraints are also added to impose a single path routing requirement. The second phase of the model is to determine a suit-

able weight set to route best effort traffic. It is applicable for optimising or expansion of operational networks running IGP/MPLS. In addition, it is also shown that non-linear constraints can be transformed into linear constraints given the problem's understanding. Furthermore, the model can be directly applied to networks with MPLS support without requiring any router or protocol modifications.

The second contribution of the chapter is an investigation to show that it is possible to redistribute best effort traffic which is congested due to bandwidth over-subscription by high priority traffic with just a few configuration changes. This is appealing for network providers to relieve congestion on best effort traffic, without running the risk of "blackouts".

The remaining sections are organised as follows: Section 5.2 outlines the two-phase model in detail. It also shows the transformation of non-linear constraints into linear constraints. Section 5.3 explains the experimental studies, results and discussion. Section 5.4 looks at the effects of dynamic changes in over-subscription and Section 5.5 concludes the chapter.

5.2 MILP / LP Model for Reconfiguration

As briefly mentioned above, the model consists of two phases. The first phase (Phase I) is a path selection process for EF and BE traffic. The second phase (Phase II) establishes a weight set for BE traffic for shortest path routing. The first subsection introduces the notation which is used throughout this chapter and also describes the Phase I MILP model and the second subsection describes the Phase II LP model.

5.2.1 Phase I: Path Allocation Model

The first phase of the model is an extended multi-commodity flow problem [10], with additional constraints. The basic formulation of the restoration model is given in [74]. In the model, high restoration percentages may yield an infeasibility condition. In this work, a variable is defined, viz: y_i that represents the "additional capacity required". This "elastic" capacity is heavily penalised in the objective function to discourage its use in the final solution.

$$\sum_{k \in K} \sum_{t \in T} \sum_{P \in P^k} \delta_i^{tk}(P) f^{tk}(P) \leq u_i + y_i \qquad \text{for } i \in E \tag{5.1}$$

Both EF and BE traffic carried on the corresponding paths should be at least restored above the restoration percentages.

$$\sum_{P \in P^k} f^{tk}(P) \geq r^{tk} d^{tk} \qquad \text{for } t \in T, k \in K \tag{5.2}$$

Restored traffic carried on the corresponding paths should not be exceeding the end-to-end demand.

$$\sum_{P \in P^k} f^{tk}(P) \leq d^{tk} \qquad \text{for } t \in T, k \in K \tag{5.3}$$

EF traffic is a special case of restoration, which can be used to simplify equation (5.2) and (5.3). In this special case, where the value of $r^{tk} = 100\%$, the equations (5.2) and (5.3) become a single equality constraint.

$$\sum_{P \in P^k} f^{tk}(P) = d^{tk}$$

It is desirable that the demands be carried on a single path. Although load balancing, (using multiple LSPs and ECMP) can be done, it is not desirable to

have packets belonging to the same OD-Pair travelling over different paths. Hence, a single path requirement needs to be imposed on EF and BE traffic. To accommodate this, equations (5.1-5.3) need to be modified as follows:

$$\sum_{k\in K}\sum_{t\in T}\sum_{P\in P^k}\delta_i^{tk}(P)a^{tk}(P)f^{tk}(P) \leq u_i + y_i \qquad \text{for } i \in E \tag{5.4}$$

$$\sum_{P\in P^k} a^{tk}(P)f^{tk}(P) \geq r^{tk}d^{tk} \qquad \text{for } t \in T, k \in K \tag{5.5}$$

$$\sum_{P\in P^k} a^{tk}(P)f^{tk}(P)d^{tk} \leq d^{tk} \qquad \text{for } t \in T, k \in K \tag{5.6}$$

$$a^{tk}(P) \text{ is binary}$$

$$f^{tk}(P) \geq 0$$

However, these constraints become non-linear constraints, due to a product of $a^{tk}(P)$ and $f^{tk}(P)$. To overcome this problem, a deeper analysis is carried out and it can be shown that the relationship between $a^{tk}(P)$ and $f^{tk}(P)$ can be written differently. Consider the following inequality:

$$f^{tk}(P) \leq d^{tk}a^{tk}(P)$$

If $a^{tk}(P)$ is equal to zero, this indicates that the path does not carry any flow at all, the value of the corresponding f must be zero. However if $a^{tk}(P)$ is equal to one, the amount of flow on the corresponding path must be non-zero and its minimum is determined by the restoration percentage and it is bounded by the d^{tk}. For a particular traffic class t for OD-Pair k, there must be only one non-zero $f^{tk}(P)$ (single path requirement).

Hence, the following linear model is seen to be equivalent to the above non-linear problem:

$$\sum_{k\in K}\sum_{t\in T}\sum_{P\in P^k}\delta_i^{tk}(P)f^{tk}(P) \leq u_i + y_i \qquad \text{for } i \in E \tag{5.7}$$

$$\sum_{P\in P^k} f^{tk}(P) \geq r^{tk}d^{tk} \qquad \text{for } t \in T, k \in K \tag{5.8}$$

$$\sum_{P\in P^k} f^{tk}(P) \leq d^{tk} \qquad \text{for } t \in T, k \in K \tag{5.9}$$

$$\sum_{P\in P^k} a^{tk}(P) = 1 \qquad \text{for } t \in T, k \in K \tag{5.10}$$

$$f^{tk}(P) \leq d^{tk}a^{tk}(P) \qquad \text{for } P \in P^k, t \in T, k \in K \tag{5.11}$$

$$a^{tk}(P) \text{ is binary}$$

$$f^{tk}(P) \geq 0$$

Since we want to maximise the throughput of the BE traffic and at the same time we must ensure that we do not use additional capacity, the problem becomes a multi-objective optimisation. The first objective is to maximise the BE traffic throughput and the second is to minimise the cost of purchasing additional capacity to accommodate the EF traffic. The third term is required to ensure that paths with a smaller number of hops are preferred.

$$Max\Big(\sum_{k\in K}\sum_{t\in T}\sum_{P\in P^k} f^{tk}(P) - M\sum_{i\in E} p_i y_i - \sum_{k\in K}\sum_{t\in T}\sum_{P\in P^k} h^{tk}(P)a^{tk}(P)\Big) \tag{5.12}$$

A large number, M, in the objective function (5.12) is introduced to discourage traffic from using the additional capacity, unless the whole network cannot sustain this level of traffic any longer.

On the classic multi-commodity flow problem, the feasibility limit of the problem is determined by the capacity of the network (i.e. the u_i values). This means that, if we slowly increase the demand vector d^{tk}, a point exists where the problem becomes infeasible. However, in this modified formulation (with variable y_i being introduced), it should be noted that the feasibility condition is not restricted by the demand level.

5.2.2 Phase II: Determining a weight set (Weight Setting)

A set of link weights needs to be determined to route the BE traffic. Work in [12] states that for any arbitrary set of routes (routing pattern in our case), as long as they are not loopy (do not contain any loops), they can be converted to shortest-paths with respect to some set of positive link weights. However, this only guarantees that the path is one of the shortest paths. A set of link weights that yields a unique shortest path, where in there is one and only one shortest path, for every demand cannot be guaranteed to exist.

A pure LP formulation [11] can be used to determine the set of link weights. Given that the routing pattern for BE traffic is known from Phase I, the problem of determining a unique shortest path weight set can be formulated as a pure LP problem as (P_n is defined as a path that carries non-zero flow based on the Phase I solution):

$$\sum_{i \in P_n} w_i + 1 <= \sum_{i \in P_j} w_i \qquad \text{for } P_n, P_j \in P^k, j \neq n, t \in T, k \in K \tag{5.13}$$

$$w_i \geq 1 \qquad \text{for } i \in E$$

The constraints (5.13) forces the path that carries the flow so that (P_n)

should be at least one unit shorter in length than other paths belonging to that OD-Pair. However, this might not always be feasible due to the reason stated above. By removing the "+1" coefficient on the left-hand side of constraints (5.13), the model becomes an equivalent method to finding "non-unique" shortest paths as in [12].

If for some reason, there are some links whose weights are not allowed to be altered, this formulation can be modified to suit this requirement. An additional constraint that says that the value of w_e for that particular link must be a known integer is simply to be added.

It should be noted that the objective function in Phase II is not important. One could use the minimising sum of link weights (as we have done in this work) or something different. Some different objectives such as minimising the range of link weights that should be used in practice or minimising number of weights that need to be changed can be used instead.

5.3 Results and Discussion

This section introduces the experimental setup, the solution procedures and provides a discussion of the results.

5.3.1 Experimental Setup

The test network consists of 8 routers that are connected by 24 uni-directional links as shown in Fig. 5.1. The demand is generated from one router to every other router in the network, hence there are 56 OD-Pairs. Every OD-Pair will have two different traffic classes, viz: EF traffic and BE traffic, in proportion 20% and 80% respectively. In this experiment, we set the restoration

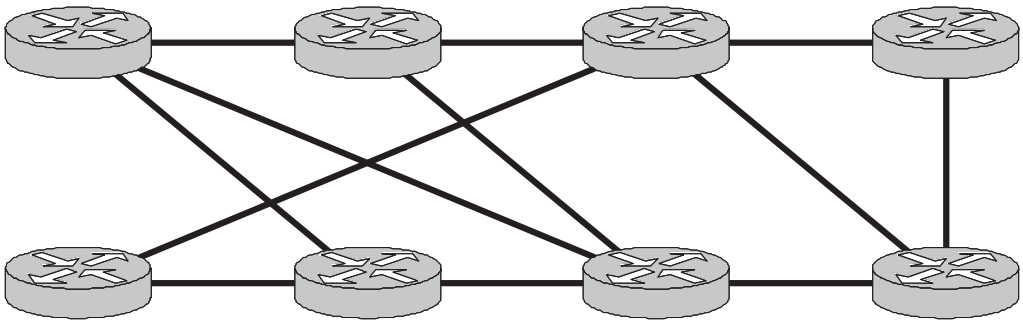

Figure 5.1: Test Network Topology

percentage for EF and BE traffic to 100% and 95%, respectively.

The experiment is done with a number of different traffic matrix instances. There are 3123 instances of these traffic matrices. They are then grouped into 41 different groups according to their total demand, with the first group having the lowest total demand and the last group having the highest total demand. Roughly, they can be categorised as light, moderate and heavy loads. Details of how these traffic matrix elements are generated are explained in the next few paragraphs.

We model the network as a set of N nodes connected to one another by a set of bi-directional links. Each link has an associated capacity, which represents the maximum amount of traffic that can be carried by that link in either direction. Paths in the network consist of an origin node, a destination node (together known as an OD-Pair) and an associated directed acyclic sequence of links that join the two nodes.

To produce the demand matrix collections we generate a set of demand scenarios consisting of vectors of $N(N-1)$ demands. The demand vectors are produced using a random traffic generator, although historical network data could also be used, if available. The random traffic generator attempts to produce demands that are sensible given the network topology and link

capacities. To this end we denote by O_i the sum of the capacities of all outgoing links connected to node i and by I_i the sum of the capacities of all incoming links to node i. In fact O_i and I_i are equal, but we differentiate them in this way to highlight their roles in what follows.

The total traffic injected into the network at node i can never exceed the value of O_i. Likewise the total traffic exiting the network at node j can never exceed I_j. If a total of T_i units of traffic enters the network at node i, it is reasonable to assume that the amount of traffic destined for node j is in proportion to the egress capacity I_j. Thus for a total traffic of T_i units entering at node i, T_{ij} units are destined for node j, where

$$T_{ij} = T_i \frac{I_j}{\sum_{k \neq i} I_k}$$

We use a network load factor $d \in [0, 1]$ to control the overall traffic load on the MPLS network by setting $T_i = dO_i$. Hence as we vary d from 0 to 1 the total traffic load goes from 0 to the maximum amount that can be injected into the network. In general however, it is not possible to find solutions with $d = 1$. The reason for this is that since the links must carry traffic for many pairs simultaneously, as d increases it becomes more and more difficult to find a solution in which a subset of crucial links is not overloaded. In fact in our experiments we have limited d to be in [0.1, 0.5445].

To ensure some randomness between the demand scenarios, each element in the demand vector is drawn from a Gaussian distribution with mean

$$i = dO_i \frac{I_j}{\sum_{k \neq i} I_k}$$

and standard deviation $8d$.

5.3.2 Solution Procedure

The MILP/LP model requires a set of paths for every demand in the network. Theoretically, all possible paths in the network should be included. However, in reality, only paths with an acceptable delay or hop counts will be used. Based on this knowledge, we can generate a "limited" number of paths. For this particular network, we generate 10 paths for each demand.

Once the Phase I problem has been formulated, it is then solved using *glpk [75], the GNU LP solver*. For this problem size, the solver is able to get within 0.1% of the optimal value in less than 5 seconds. We use *glpk*, which is a free version of LP solver, as opposed to the commercial one, CPLEX. One of the reasons for using glpk in this case is to check problem complexity; ie whether the MILP problem is too large for reasonable operations.

Once the Phase I solution has been obtained, the Phase II problem can be formulated. A similar approach to that described in [11] can be adopted. To solve the Phase II problem, one can start with only two shortest paths and do an iterative computation. In the first iteration, the first path, P_n is the one that carries the flow and has the total metric less than P_j, where $j \in J, j \neq n$. Once the problem is solved, Dijkstra's algorithm can be used to determine whether there are more shortest paths that are not included in the Phase II problem. If there is one, then it will be added to the Phase II problem and it is solved again. If the inclusion of the new path causes infeasibility, this indicates that no weight system exists for this routing pattern. The iteration stops when no more shortest paths can be found.

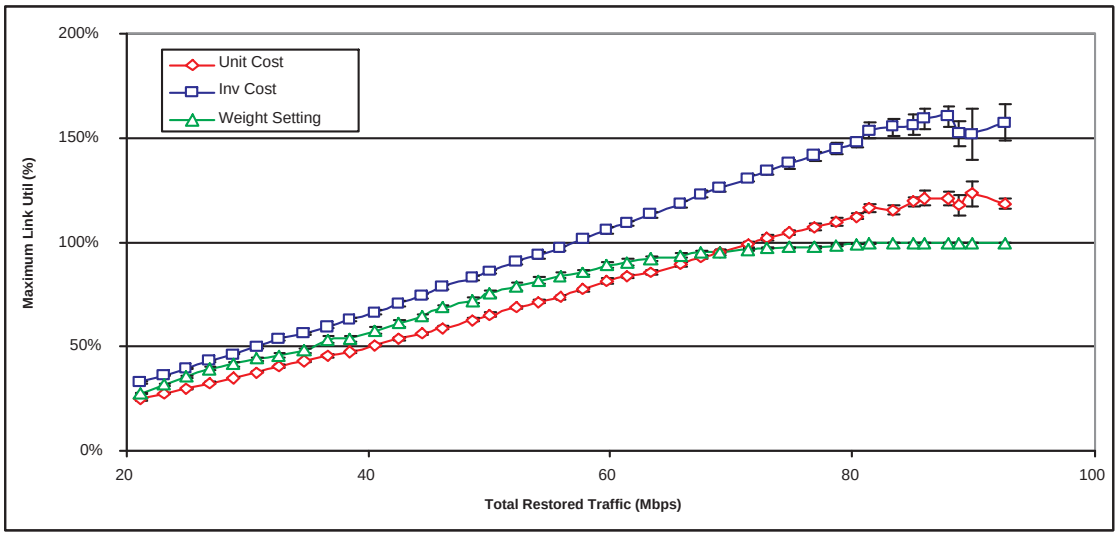

Figure 5.2: Resulting Maximum Utilisation for BE Traffic Routing

5.3.3 Performance Evaluation

In this work, the concern is BE traffic. EF traffic is routed using MPLS and has a guaranteed minimum bandwidth. Moreover, once the solution from the Phase I problem is available, EF traffic can be readily routed. Our main concern here is the maximum utilisation of the links in the network.

Figure 5.2 depicts the resulting maximum utilisation when BE traffic is routed, based on three different routing schemes, namely, unit weight, inverse capacity and weight setting (based on the Phase II solution) with increasing total demand. The error bars on every data point indicate the 95% confidence limits obtained from the simulation.

Using unit weights in this simple network, will generate many multiple shortest paths for most of the OD-Pairs. In this network, the splitting mechanism is effective in spreading the load throughout the network. In a real network running OSPF, Equal Cost Multi Path (ECMP) functionality is used to split aggregate traffic to a particular destination evenly.

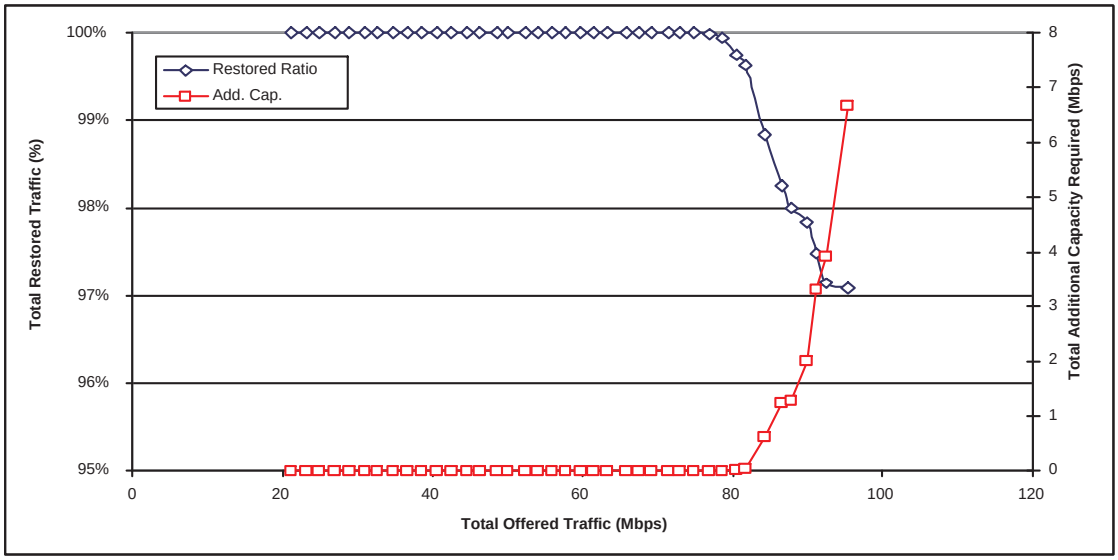

Figure 5.3: Percentage of traffic restored and additional capacity required

Using the inverse capacity as the shortest path routing metric gives the worst performance of all. This is true across different network loads in our experiments. The utilisation reaches 100% when the average demand is 57 Mbps. With the unit weight and weight setting, 100% utilisation is reached when the total demand is 72 Mbps. Furthermore, due to the capacity constraints in the model, the weight setting can restrict the maximum utilisation at 100%, whilst the other results give 120% and 155% (highest traffic load). Variation in the maximum link utilisation in the Unit Weight and Inv Cap cases is higher because none of these models have a capacity restriction.

Although the Unit Weight approach may look as if it performs reasonably well in comparison to weight setting, a complication with multi-path routing arises because it is impossible to split traffic evenly in general practice. Splitting can degrade TCP performance because packets belonging to the same OD-Pair may be arriving out of order (packet level splitting). Splitting can also cause problems in debugging network problems. Test packets can travel

via several different paths and conceal problematic paths.

One of the goals of the Traffic Engineering process is to delay any required capacity expansion, although the traffic demand is growing without performance degradation. Herein, using weight setting, we show that capacity expansion can be delayed for a while. Weight setting can maintain feasibility without having have to put in additional capacity until the total demand reaches 84 Mbps (see Fig. 5.3 the red curve). When it is required, we also show that the amount is very small in comparison to the demand that can be accommodated. The model is able to identify the bottle-neck links in the network. Upgrading these links will greatly increase BE traffic throughput.

The blue curve in Fig. 5.3 shows the fraction of total BE traffic restored to the total BE traffic offered. Recalling that the BE traffic restoration percentage is set to 95%; in all the scenarios, the percentages are well above the target restoration percentage. A long OD-Pair (the one that has a significant number of hops between the source and the destination) will be just restored based on the restoration percentage lower bound (in this case 95%) because of network bottle-necks. A short OD-Pair (one that only has one or two hops to travel to the destination) will benefit greatly because they will be restored up to 100% on the "non bottle neck" links.

In practice, it is impossible to have 100% utilisation or higher. Utilisation figures in this work are to be normalised with the utilisation of a running network. It should be noted that the utilisation results depicted in Fig. 5.2 are calculated after the additional capacity has been added. Fig. 5.2 and 5.3 are the experimental results after the data points are grouped together based on their total demand. For the "un-grouped" results, please refer to Fig. 5.4 and 5.5.

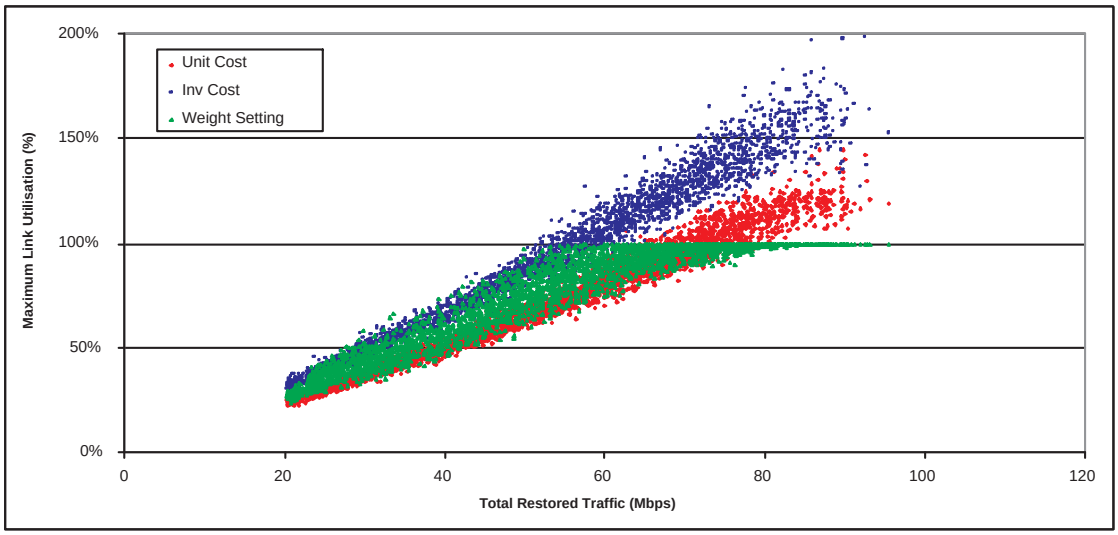

Figure 5.4: Resulting Maximum Utilisation for BE Traffic Routing

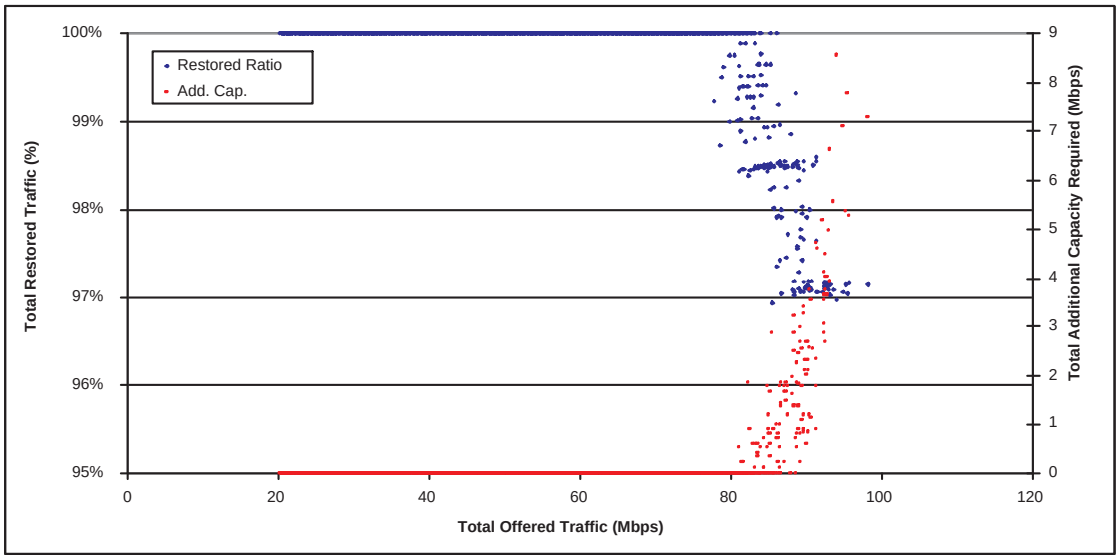

Figure 5.5: Percentage of traffic restored and Additional Capacity required

5.4 Reacting to Dynamic Changes in Over-subscription

During network operation, traffic demand changes in a dynamic way. A network administrator may be required to provide more resources for high priority traffic or customers terminating their leased line might also be the reason

for a traffic demand change. Regardless, all of this can cause the network to be reconfigured. In particular, the possibility of the resources of one class being under-utilised while those of another class being over-subscribed and thus impacting on service levels leads to the need for dynamic resource management [49]. Also the use of dynamic resource management will increase the frequency with which a network service class will require re-optimisation. This amplifies the case for a fast calculation technique able to remove link "hot-spots" caused by resource changes with minimal disturbance to the network.

Firstly, we (somewhat arbitrarily) assume that the maximum permissible link utilisation (before queueing delay becomes prohibitive) is 48%. We also assume that the average total of best-effort and higher priority traffic is roughly constant. We now optimise the network assuming the maximum permissible link utilisation is 40%. If we now place high priority traffic on an explicit route in the network, it means that the residual capacity for best-effort traffic is reduced and certain links may become "hot spots". To remove these hot spots we must re-optimise the network. If we re-optimise from an "operating point" of 40%, we observed considerable network disturbance because there is no "network slack" and flows cannot be re-routed easily. To avoid this problem, we re-optimise from an operating point of 48% (see Figure 5.6 and 5.7). Note that we have used the non-linearity of these curves to greatly increase the "network slack" with only a small increase in maximum permissible utilisation. Also, since the number of non-zero dual prices is now only twelve, from the complementary slackness conditions [10], we see that only twelve links have a link utilisation of 48%. We can now re-optimise quite severe hot spots very quickly with very little disturbance to the network.

These experiments are conducted on a 56 node network. In the first ex-

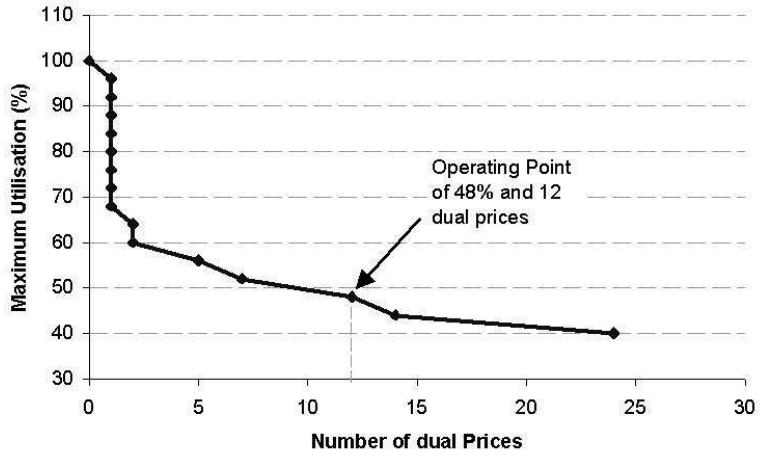

Figure 5.6: Re-optimisation on 56 node network - Random link capacity and Cisco default metric

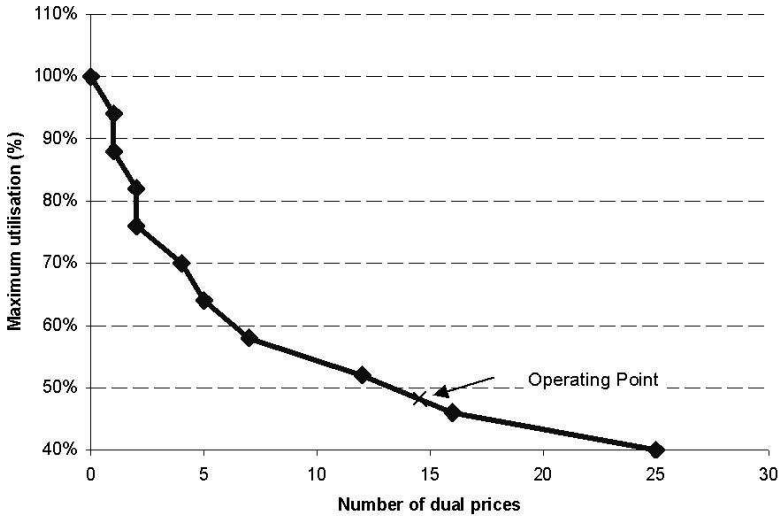

Figure 5.7: Re-optimisation on 56 node network - Uniform link capacity and random metric

periment, the link capacities are drawn from a uniform distribution and the routing metric is set according to the Cisco manufacturer default, namely, the metric is set to the inverse of link capacity. In the second experiment, the network has uniform link capacity. Having an inverse link capacity is equally the same as having a unit weight. As stipulated in the previous chapter, having unit weights will create many equal cost paths and degrade the re-

optimisation performance. Hence, we want to avoid this as much as possible. Randomising the initial metric is one of the ways of doing this. It should be noted that the optimisation model for two classes of traffic that is presented in the first part of this chapter produces a single path routing pattern, which makes it a suitable input metric for this re-optimisation technique.

5.5 Conclusions

In this chapter, we have presented a model to reconfigure Diff-Serv networks with two classes of traffic. MPLS technology can be used to route the high priority traffic, whereas native IP routing, such as OSPF, is used to route the best effort traffic. The model maximises the best effort traffic throughput and minimises the cost of expansion, whilst at the same time fulfilling bandwidth requirements for high priority traffic. It is shown that in this particular model, non-linear constraints can be transformed into linear constraints. Experimental results show that the model can reduce the maximum link utilisation in the network by as much as 50% and accommodate 44% more traffic without needing capacity expansion.

Heuristics to solve MILP model could be developed to solve larger network problems. For large networks, general purpose LP solvers usually run very slowly and may not find any solutions. Future work includes the comparison of a heuristic method with the MILP solution generated from LP solvers for large networks.

We also show that under this model, our approach is able to optimise the remaining best-effort traffic and further, that it is able to very quickly return a network to near-optimal operation after a new high-priority flow is placed. Thus, our technique provides effective dynamic resource management.

Chapter 6

Comparison of Traffic Engineering Methods

This chapter presents a comparison study of two Traffic Engineering (TE) methods for optimally routing flows in an autonomous system (AS). The first method is based on link weight setting and the second method uses MPLS Explicit Route using the Neural Network - Marginal Increase Heuristic [76]. The comparison study is done by using the well-known ns-2 simulator. To the knowledge of the author, there are no other existing works addressing the performance comparison between these TE methods. Furthermore, these two methods are novel compared to existing methods for they are able to provide TE solutions in the order of seconds for medium sized networks. We could always argue that better solutions can always be found given more time. However, the emphasis here is that both methods are made suitable for online TE. It is also shown here that both methods greatly outperform the commonly used IP routing protocols, such as default OSPF. This is a collaboration work with Telstra Research Laboratories.

6.1 Introduction

As previously observed, TE has received growing interest by Internet Service Providers (ISPs) because it allows efficient use of existing network resources. As a result of TE, networks can sustain more traffic, give customers an improved end-to-end network service and help ISPs to meet Service Level Agreement (SLA). It is very important that ISPs deal with congestion issues in a timely manner such that they don't break the SLA commitments with customers.

However, there are only few methods that are suitable for carrying out online TE functionalities. Most of the search based techniques are too slow to provide an online solution, a solution that is available in a matter of seconds is what is required in this case. We have shown in a previous chapter that LP-based weight setting can give a solution in merely seconds for shortest path type networks.

Nowadays, ISPs are migrating their networks to MPLS enabled networks. TE in MPLS networks can be carried out in two different ways. The first is to perform a constrained shortest path computation for every flow arriving at the edge of an MPLS network. This is a short term form of TE. It can be employed if the network is rather small and it does not carry thousands of flows. The second is to employ fixed explicit routing for flows inside the network. The problem of MPLS TE then becomes how to select a suitable path that can accommodate all the requirements for every OD-Pair. Due to the desire for single path routing in MPLS networks, the problem becomes a Mixed Integer Linear Programming / MILP problem. Hence an optimisation process to obtain a TE solution for MPLS networks typically requires a longer time to solve.

General MILP solvers usually solve a problem using a two-stage approach. The two-stage approach consists of the LP-relaxation phase in which the integer constraints are ignored and the pure LP problem is solved. This is then followed by a subsequent search for a solution near the pure LP optimum, while satisfying the integrality constraints. It is the second stage that often leads to implementing an MILP strategy to explore the solution. Whilst the LP-relaxation may have a feasible solution, a solution to the corresponding MILP problem may not exist. Furthermore, even if it exists there is no guarantee that it can be found within a given "reasonable" time frame. Hence, having a technique than can speed up the MILP solution search is desirable. By providing a good initial solution for MILP problems, the NN-MIH system improves the solution time. It has been showed in [76], that using a trained neural network (NN) to obtain an initial solution. A heuristic technique, marginal increase heuristic (MIH), is provided to fix up the broken constraints. Overall, the combination of NN-MIH can speed up the MILP optimisation such that the solution can be attained in order of seconds.

The contribution outlined in this chapter is the performance comparison of two TE methods, namely the LP-based Weight setting [55] and the NN/MIH [76] method for MPLS path selection. It introduces simulation results that demonstrate the performance in terms of packet loss for the two TE methods and compares them to the standard shortest path routing common in IP networks. The experiments are done by using the popular simulator ns-2. The performance parameter is the packet loss probability as a function of increasing traffic demand.

6.2 Optimisation Methods Overview

This section introduces a brief overview of the optimisation methods.

6.2.1 LP-based Weight Setting

Intra-domain routing protocols such as Open Shortest Path First (OSPF) or Intermediate System-Intermediate System (IS-IS) compute the routes to the destinations based on link metrics (weights). Links that have a lower weight are treated by the routing protocols as preferable routes. Hence, these links are more prone to congestion. Congestion management, in these cases, can be achieved by employing routing protocol weight setting. Fortz and Thorup [6] suggested that changing a few link weights can re-balance the link loads in the network. In this thesis and in [55], it has been shown that a solution of the weight setting problem can be obtained in the order of seconds by using an LP formulation.

Using the complementary slackness conditions for the path-flow formulation of the Multi-Commodity Flow (MCF) problem from [10], a set of "modified" link metrics (the original OSPF metrics plus the non-zero link dual prices) can be obtained such that the optimal paths are shortest paths with respect to these modified metrics, for a given traffic matrix. These modified link metrics are then used as the OSPF weights. Earlier results and more details can be found in Chapter 3.

6.2.2 Neural Networks/Marginal Increase Heuristic

MPLS allows network administrators to re-direct traffic streams on selected paths via its explicit routing capability. The traffic engineering process is

achieved by judiciously selecting these paths such that every traffic stream receives a good service. In the literature, the optimal path selection problem is formulated as a Mixed Integer Linear Programming (MILP) [77]. Most often, the integer constraints are used to represent practical requirements of the network architecture, protocols or management systems. For example, bandwidth must be allocated in multiples of a distinct set of link capacities or flows to be restricted to a single path through the network.

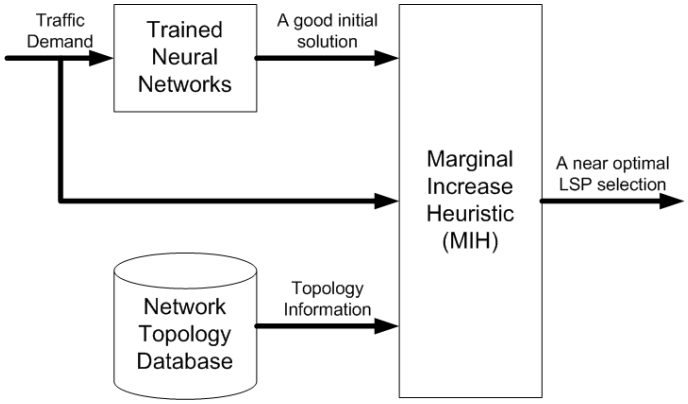

Figure 6.1: Neural Network - Marginal Increase Heuristic (MIH) system

The solutions, which are then used as training data for NN, must conform to all the physical constraints of the network as well as the offered traffic constraints. The working principle of MIH is to take the initial solution, which is provided by the NN and to rectify any flaws due to broken constraints. For details see [76].

Generating Training Data for Neural Networks

A model for single path traffic engineering problem, which aims at load balancing and minimisation of resource usage, is proposed in [77]. The resource usage minimisation objective is to prevent the OD-Pair to take a longer path in the network. However, based on our experiments in the network test-bed,

the resource usage objective has proven not to be effective to lower the packet loss probability; thus it is not taken into consideration. The simplified model can be described as in equations (6.1) - (6.5).

$$max \qquad B \tag{6.1}$$

$$\sum_{k \in K} \sum_{p \in P_k} \delta_{pe} d_k u_{kp} = \zeta_e u_e \qquad \text{for all } e \in \mathbf{E} \tag{6.2}$$

$$B \leq 1 - \zeta_e \qquad \text{for all } e \in \mathbf{E} \tag{6.3}$$

$$\sum_{p \in P_k} u_{kp} = 1 \qquad \text{for all } k \in \mathbf{K} \tag{6.4}$$

$$u_{kp} \text{ is 0 or 1} \qquad \text{for all } k \in \mathbf{K} \text{ and } p \in \mathbf{P} \tag{6.5}$$

The objective function chosen is to maximise the so-called balance factor, B, a fraction of spare capacity on the most heavily utilised link (eq. 6.1). It is computed from all link utilisations, ζ_e, in the network (see eq. 6.3). A soft capacity limit (see eq. 6.2) is introduced to calculate the amount of traffic on a link. The single path routing is also imposed by introducing binary decision variables, u_{kp}, and constraining those which belong to the same OD-Pair to have a sum of one (eq. 6.4).

For each traffic demand scenario, a conventional MILP solver is used to provide an optimal solution for a given network topology. An instance in the neural network training set consists of a pair of a traffic matrices and a set of path configurations determined by the MILP solver for every OD-Pair. The neural network is trained to associate a particular path configuration with a particular traffic demand matrix. Figure 6.2 outlines the process of generating the training data.

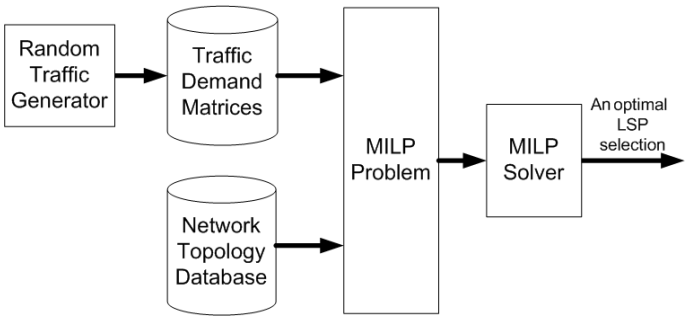

Figure 6.2: Training data for Neural Networks

Marginal Increase Heuristic (MIH)

Due to the nature of neural networks, the initial solution produced might not fully conform with all MILP problem constraints. This results in a few violations of some of the constraints that need to be enforced. The role of the MIH is to refine the initial solution provided by the NN and correct any flaws due to unsatisfied demand or other broken constraints. The operating principle behind MIH is to select an OD-Pair and a corresponding path and allocate capacity to that OD-Pair and path adding one unit at a time.

Initially, it checks the initial solution to see if all OD-Pairs have their demands satisfied. If this is the case, then the solution is valid and the MIH process is not required. Otherwise, the MIH selects the OD-Pair and the path allocation such that there is only a small increase in the objective function. This process continues until either all OD-Pairs have their demand requirements satisfied or there are no more paths capable of transporting an extra unit of demand.

The ordering of the paths should be chosen to have their capacity determined by their cost. The link cost is defined as an increase in the link utilisation as a result of adding one more unit of traffic. The path cost is computed by finding the maximum of the link costs in that path if this path does not

yield a reduction in the objective function, B is a fraction of the spare capacity on the most heavily utilised link, in the MILP formulation. Otherwise, the path is highly penalised making it less favourable for carrying traffic [76].

6.3 Experimental Setup

A series of simulations on two different network topologies was carried out to benchmark the performance of the default OSPF routing based on Cisco's recommendation, the optimised routing based on LP-based weight setting and the NN/MIH MPLS method. We compared the performance of the three routing schemes based on their packet losses as a function of increasing total demand.

For the first set of simulations, we selected a network which consisted of 23 nodes, connected by 86 uni-directional links as depicted in Figure 6.3. The link capacities ranged from 512 Mbps to 16,384 Mbps. Cisco recommends that the link weights should be set based on the inverse of the link capacity. The link latency is chosen to be 5 ms. We assume that the traffic represents aggregated flows from every origin to every destination in the network. Thus there are 506 individual flows which need to be accommodated, where each flow represents the offered traffic for an OD-Pair. Each flow is modelled as a constant bit rate source sending 500 byte packets from the origin to the destination. The interval rate is determined by the OD-Pair demand which is randomly generated using a Gaussian distribution with a specified mean and standard deviation. The simulation length is 3 seconds including 0.1 seconds for the warm-up period. The statistics collection starts after the warm-up period has elapsed.

The simulation is done across a number of different traffic matrix instances.

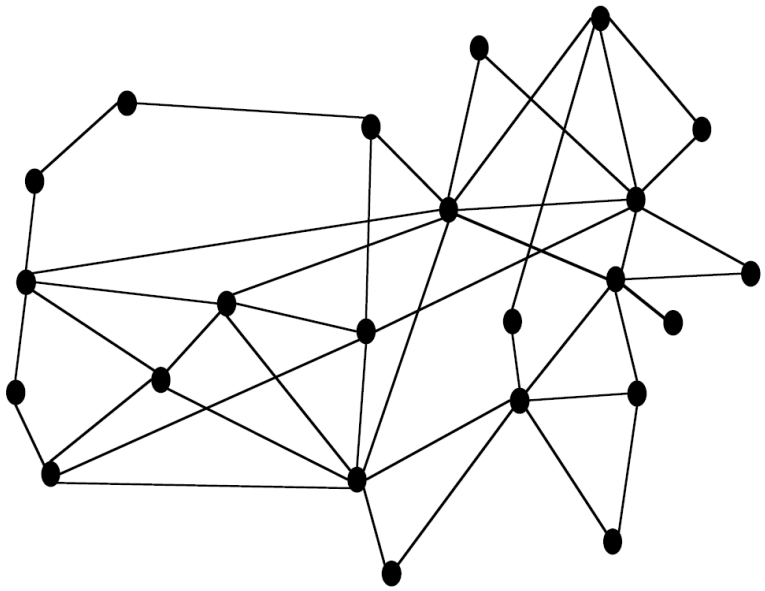

Figure 6.3: The first test network topology - a 23-node network

There are 374 instances for this experiment. There are 506 traffic elements in each of the instances. Each of these elements is generated based on the node's activity as a sender and a receiver. To ensure some randomness, traffic elements are drawn from a distribution as described in [76]. A node with high capacity links originating from it will generate more traffic. Conversely, a node with high capacity links terminated at it will sink more traffic. In practice, the traffic matrix is obtained using historical data, direct traffic measurements, estimation techniques [78] or a combination of these techniques.

The inputs for these optimisation techniques are as follows: the source and the destination node of the OD-Pair and its associated demand, the network topology and the link capacities. This information is then used to formulate the LP weight setting problem. The LP solver, CLPEX, determines the dual solution, which is then used to modify the OSPF metrics. This process is repeated for each of the 374 traffic matrices (instances) and each will have a different set of weight changes.

To obtain training data for the NN, the path selection problem is solved using the MILP solver. The solution is a series of MPLS paths for each corresponding OD-Pair for a given traffic matrix. The same process is repeated over a number of traffic matrices to give sufficient training data. Once the NN training is completed, this NN can generate an initial solution for the MIH. For each of the 374 given instances, these initial solutions can be obtained by the NN and they are then refined by the MIH. Each will have an MPLS path or LSP to route the demand.

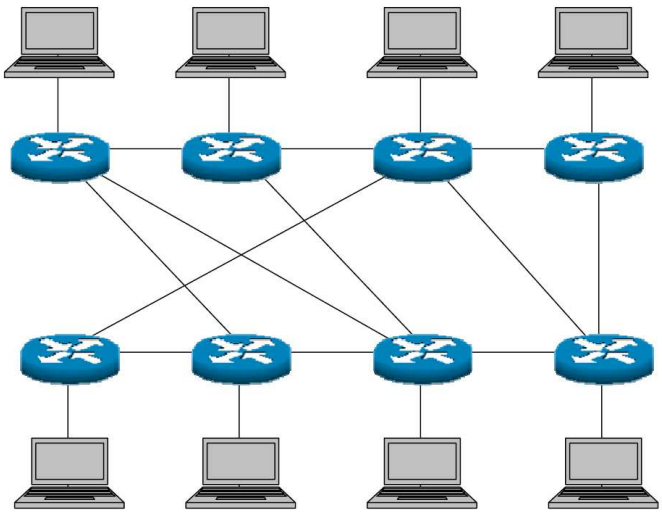

Figure 6.4: The second test network topology - a 8-node network

The second topology that we simulated consisted of 8 nodes and 24 unidirectional links. There were two types of links, 10 Mbps and 100 Mpbs. There were 3887 traffic matrices that were simulated with this topology. The elements of these traffic matrices were generated in the same fashion as that of the 23-node simulation.

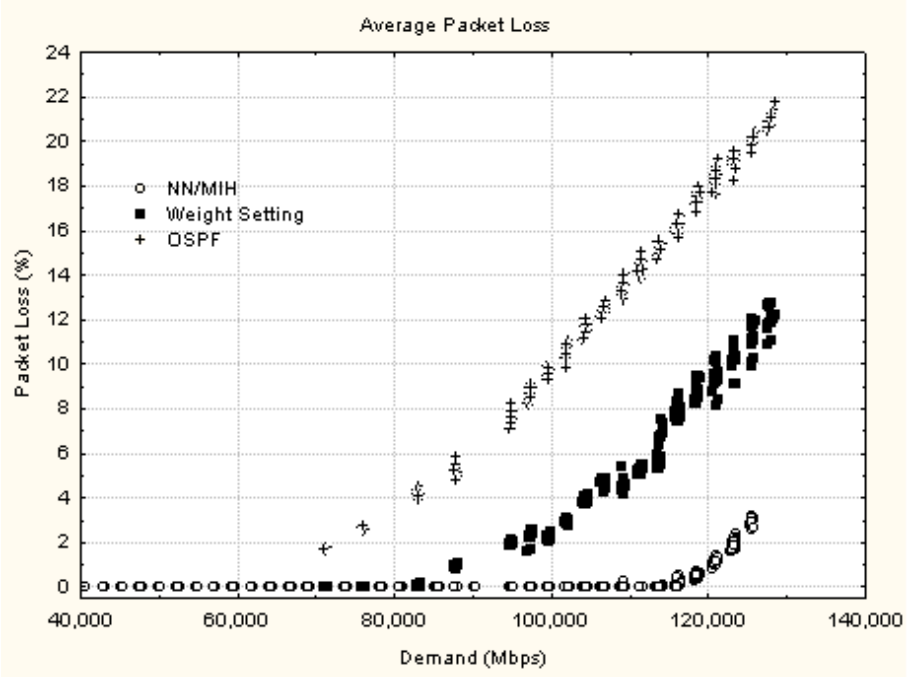

Figure 6.5: Average OD pair loss across different traffic matrices

6.4 Results and Discussions

In these experiments, the parameters of interest are the average packet loss and maximum packet loss across all OD-Pairs that exist in the network. The average packet loss is computed by averaging the loss probabilities for all 506 OD-Pairs. The maximum packet loss is taken as the maximum figure across 506 OD-Pairs. These figures are plotted as a function of the total demand in Mbps in Figure 6.5 and 6.6.

Figure 6.5 and 6.6 depict the average OD-Pair loss probability and the maximum OD-Pair loss across different traffic loadings. Both TE methods extend the break away point (a point where packet loss starts to occur) compared to OSPF routing significantly. With an increasing load, it is expected that the network would start to experience congestion, which results in a number of packets being dropped. Simulation with the OSPF default metric starts to ex-

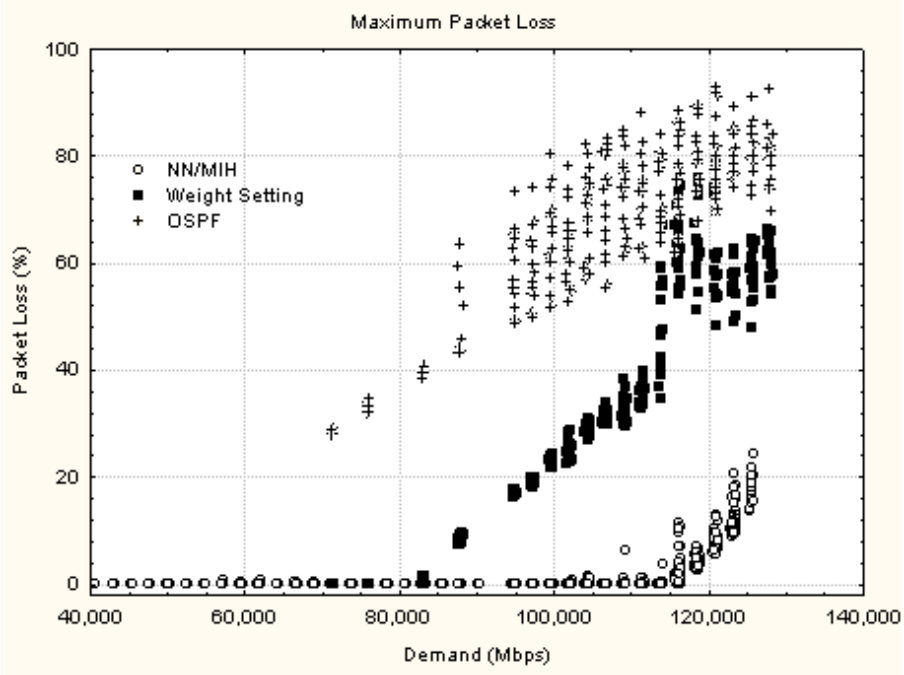

Figure 6.6: Maximum OD pair loss across different traffic matrices

perience packet loss when the total demand approaches 70 Gbps (see Figure 6.6). The break away points for weight setting and NN/MIH are 88 Gbps and 109 Gbps. These indicate that the network can sustain at least about 26% and 56% more traffic respectively.

There are artifacts in the simulations where traffic is routed based on the link metrics (ie. OSPF and weight setting OSPF). This is shown by the wide spread in the maximum packet loss (see Figure 6.6). The ns-2 simulator chooses a random outgoing link whenever there are multiple equal length paths from a node to a destination, rather splitting traffic through the ECMP functionality. We believe that ns-2 does not have a proper implementation of ECMP. Hence, it is not possible to guarantee that traffic is not routed on an already-congested link or path (since the outgoing link is chosen randomly).

LP-based weight setting and NN/MIH perform in a superior manner compared to the default OSPF weight setting approach across all groups. How-

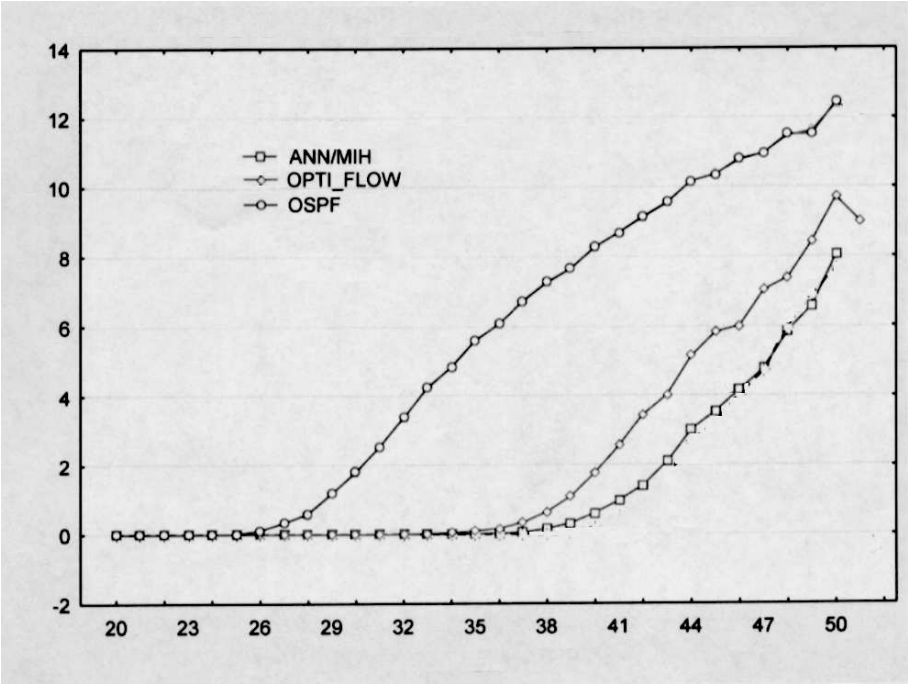

Figure 6.7: Average OD pair loss across different groups of traffic matrices on an 8-node network

ever, the advantage seems to be slowly diminishing (i.e. the two curves come closer together and to the OSPF curve) as the load increases further. At this point, the network is approaching its saturation point, where there is not much room left to reroute the traffic. When there is no more room to reroute the traffic, none of traffic engineering methods would work effectively.

Simulations were also carried out with the second network topology containing 8 nodes. However, 3887 data points corresponding to 3887 traffic matrices were too many to be plotted on one graph, so once again they were grouped according to their total demand into 31 different groups. The average and maximum OD-Pair loss probabilities were also of interest. The average and maximum OD-Pair loss results are given in 6.7 and 6.8. The X-axis denotes the increasing offered load, while the Y-axis denotes the loss probability. The results are not surprising and, once again, both TE methods perform

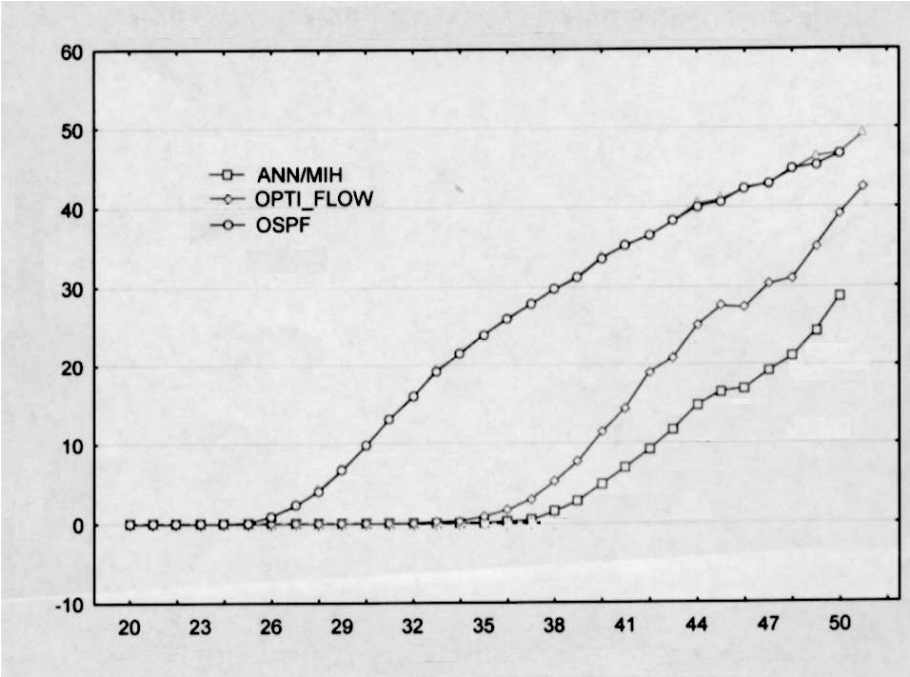

Figure 6.8: Maximum OD pair loss across different groups of traffic matrices on an 8-node network

much better compared to OSPF default routing. LP-based weight setting and NN/MIH allows more traffic carried in the network without experiencing any packet loss.

6.5 Conclusions

In networks running a shortest path based protocol, a weight setting technique can be readily deployed without the need to upgrade the existing network infrastructure. However, fine-grained traffic flow control cannot be achieved. Weight setting affects flows at the aggregate level. On the other hand, MPLS gives network administrators full control of routing traffic flows at all levels of granularity. When many equal cost paths exist in a network, it appears that MPLS explicit routing is more suitable for TE purposes. It allows aggregate

traffic, or even a single traffic stream, to be routed on a specific path. Although some automated processes can help in configuring MPLS paths, there is a fair amount of configuration required when networks are setup for the first time. Furthermore, for TE purposes, it is necessary to enumerate a sufficient number of candidate paths to be used in the solution. This can lead to an exponential increase in the number of paths for growing problem sizes.

The significance of these two TE methods is their ability to provide TE solution in real-time. Both methods provide a significant improvement in the amount of traffic that can be carried by the network over standard OSPF routing for a given packet loss probability.

Chapter 7

OptiFlow - A Capacity Management Tool

Managing large IP networks requires understanding about current traffic flows, traffic routing and network configuration (topology). However, network operators still continue to perform manual fine tuning of each IP router based on limited measurement data. The networking industry is lacking a system that can be used to support network traffic management and deliver effective traffic engineering.

To deliver a better service for network users, the service provider must provide sufficient bandwidth to carry the users' traffic. On the other hand, the service provider must also minimise their capital and operational expenditure (CAPEX and OPEX). These are often two conflicting interests. it is possible to achieve a balance by utilising all existing network infra-structure using traffic engineering solutions. Capacity expansion, regular maintenance and software updates might disconnect parts of the networks or cause network congestion if they are not managed carefully. It is also evident that traffic demand shifting or unexpected traffic fluctuations may cause network

congestion. These scenarios further support the need for a good network management system.

This motivated us to develop a software package for managing the performance of IP networks at CATT research Centre. The software package is called OptiFlow. The target users of this package will be ISP network operators. The key idea behind OptiFlow is to generate global views of the managed network on the basis of configuration and measurement associated with the network elements. Using OptiFlow, a user can experiment with changes in network configuration in a simulated environment rather than the operational network. The tool also provides the necessary framework for a network optimisation module.

7.1 Introduction

Recent years have seen a tremendous growth in Internet traffic. This on-going development increases the need for Internet Service Providers (ISPs) to operate and manage their networks effectively to avoid congestion for their customers' traffic flows. However, native IP networks have limited traffic management capabilities. This is particularly true for Open Shortest Path (OSPF) type networks. Unfortunately, ISPs only have a limited number of software systems and tools to support the management aides in measurement and control activities [5]. Network managers and operators have limited knowledge of network dynamics from a global perspective. There may be many measurement and monitoring systems for *individual* links or routers, but only very few systems permit a more holistic view.

Without global knowledge of measurement data and proper routing controls, traffic engineering solutions may be useless. This is made worse by

rapidly changing conditions of IP networks, such as congestion and failures. Link congestions can occur in these networks and is usually caused by network component failures, scheduled maintenance, traffic shifts or incorrect configurations. In these changing environments, connectivity is maintained by routing protocols that automatically compute and update routing tables at set time intervals. Although this restores connectivity, there is no guarantee that service quality is maintained. Some links might be congested because they have to carry additional traffic from the failed links, for example.

This chapter describes OptiFlow, a unified set of software tools for traffic engineering of IP backbone networks. The key idea behind OptiFlow is to generate a global view of the network components based on network discovery and data associated with them. After creating an appropriate global view of the network, we are able to visualise the network-wide implications of local changes, such as traffic shifts and routing policy changes. OptiFlow provides a platform for network managers to carry out "what-if" investigations before committing changes to the real network. In addition, the tool provides an additional network optimisation module through the traffic engineering technique described earlier.

To illustrate the capabilities of OptiFlow, we focus on one specific application: the problem of finding a set of weights to be used for shortest path routing metrics to remove the rapid build-up of congestion in intra domain networks. The overall task of finding a set of weights can be divided into several independent sub-tasks. By using topological information and usage history data, OptiFlow identifies the utilisation of each of the links and identifies the most heavily loaded link. Then, OptiFlow executes a traffic matrix estimation process and tries to identify OD-Pairs that are responsible for the congestion. Using information from the traffic estimation, OptiFlow enables a

network manager to have the capability of changing the configuration of intra-domain routing (by changing OSPF routing metrics), to direct some of these traffic demands away from congested links, and hence distributing the load more evenly in the network. OptiFlow then recomputes the routes and link loads based on the new routing pattern and provides an estimate of how the traffic would flow after the changes are invoked. The user then has the choice of accepting or rejecting the configuration changes having been informed of the estimates for the resulting load.

The contributions of this chapter are twofold: Firstly, it outlines the implementation of OptiFlow, a prototype network capacity management tool. The LP based weight setting method has been implemented and integrated into OptiFlow. Its aim is to combat link overloads and hot spots. Users can also build their own topology, set appropriate configurations and use OptiFlow as a platform to investigate what-if scenarios, e.g. investigating the effect of pre-planned maintenance on the maximum link utilisation. The second contribution is a development of a virtual network to test the implementation of OptiFlow using some open source routing software known as Zebra [79].

This short chapter is structured as follows: Section 7.2 describes the OptiFlow system architecture and briefly describes its implementation. Section 7.3 outlines the basic components of a virtual network that can be used to emulate a real OSPF network. Section 7.4 describes the experimental results after running OptiFlow in a real and virtual network. Finally, Section 7.5 concludes the chapter.

7.2 Overview of OptiFlow

OptiFlow is a Microsoft Windows application that manages OSPF networks. It is designed to read network information and configuration based on information available in the MIB (Management Information Base) database of routers in the network. A MIB table maintains a separate section for OSPF as defined in [80]. Based on an individual router's routing table, it does a recursive depth first search to discover all the OSPF enabled routers in the "domain". The domain refers to a whole ISP network or a subset within the ISP's network. The routers that will be discovered are the ones that are accessible within the user's administrative privileges.

After OptiFlow discovers the network topology, all router configurations and traffic flow information, the topology will then be displayed graphically on the tool's display canvas. The data can also be saved into a workspace file for later use. We can then take this workspace "off-line" to a simulated environment, where we can change the configuration without changing anything in the real network. We call this the capability to play "*what-if*" scenarios.

OptiFlow is also a network monitoring tool. It monitors the utilisation of the links in the network. The user can specify the utilisation thresholds that signal the onset of link congestion. Once a link utilisation crosses this threshold, the link will be coloured red on the canvas, and OptiFlow prompts the user as to whether he / she wants to invoke the traffic engineering module to optimise the network using weight setting. OptiFlow also enables the user to manually tune the OSPF link metrics.

In OptiFlow, we use the techniques described in Chapter 3. The input requirements for each of these techniques are the network topology, the link capacities and all the origin to destination pair (OD-pair) traffic demands. The

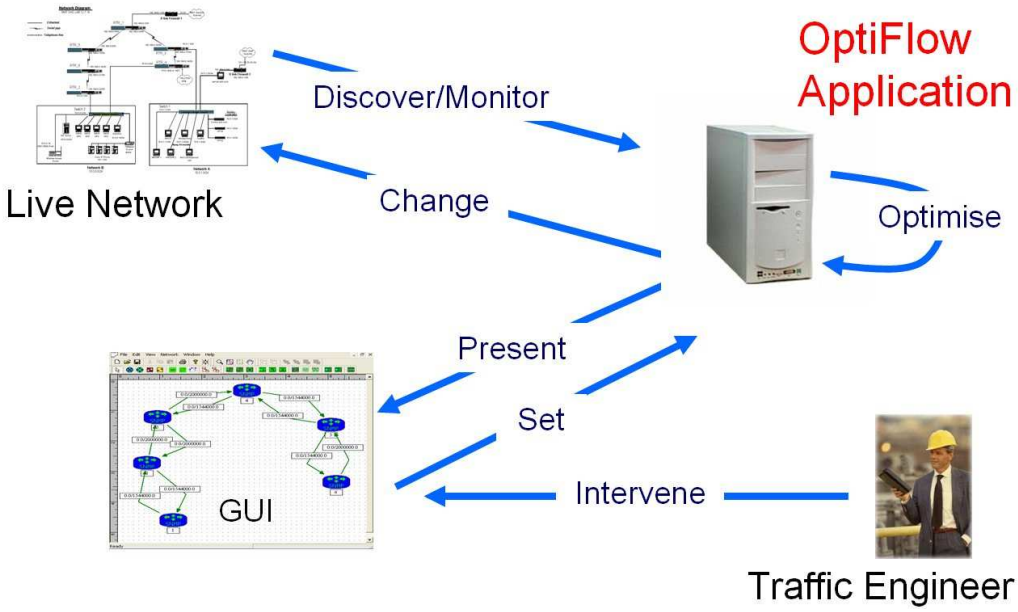

Figure 7.1: OptiFlow Traffic Engineering Scenario

first two can be obtained from network configuration information and data available from the MIB tables. However, the third set of data, the OD-pair traffic demands, is not directly available. It is possible to use a direct measurement method to determine the amount of traffic flowing between each of the source-destination pairs or use an inference method based on link utilisation measurements. Such inference methods have been of great interest in the Internet research community and some typical examples are found in [81] and [82].

Once all the inputs are available, it is possible to find a new set of link metrics that reduces the maximum link utilisation in the network. A user will be prompted with options as to whether they should change the link metrics, as suggested, or to ignore these suggestions. The user is also able to take this network to a "simulated" environment to see the resulting effects of any weight changes. Once the user approves the changes, a new set of link metrics can be distributed by the tool throughout the network (provided that the user

has appropriate authorisation to modify the MIB tables in their AS region). The user is also able to change the link metrics manually, based on his or her opinion as to how to re-route traffic away from a congested link. Figure 7.1 illustrates a scenario where OptiFlow monitors the network for congestion and prompts the user for intervention.

Due to the complexity of the overall traffic engineering problem, in OptiFlow we focus on only intra-domain traffic engineering. Research in inter-domain traffic engineering is considered as an avenue for future directions for this research. Currently, only the OSPF protocol is considered by OptiFlow. Another commonly used intra-domain protocol is IS-IS. IS-IS routing is also based on shortest path routing. Hence, the traffic engineering method that has been implemented is also applicable to networks running IS-IS as well, with minor changes of information extracted from the MIB database.

OptiFlow's system diagram is given in Figure 7.2. The OptiFlow implementation involves a network interface, network engines and steering. The Network Interface module handles all communications between the network and OptiFlow. The communication is performed using the SNMP (Simple Network Management Protocol) standard. The engine is at the heart of OptiFlow, where monitoring, measuring, optimising and controlling activities take place. It also handles the data model for the network in the Network Topology sub-module. The steering module allows a wide range of interactions to be performed with the user.

7.2.1 Network Interface

This module is responsible for handling the communications between the managed routers and the OptiFlow application. The data extracted from the

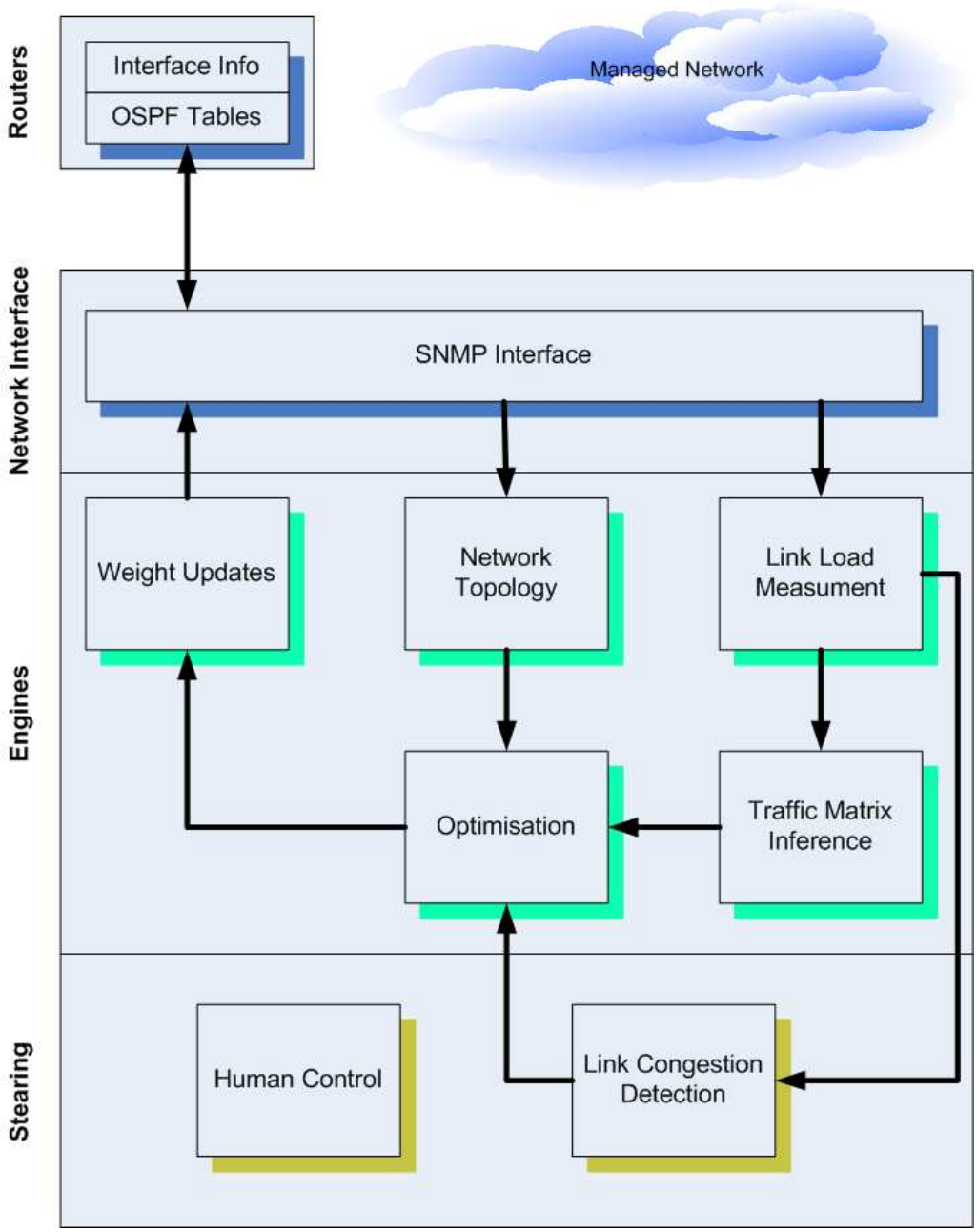

Figure 7.2: OptiFlow System Diagram

routers includes the router status, their interfaces, the OSPF routing table and link load measurements. In the reverse direction, OptiFlow sets the network configuration based on the users' input or the output from the traffic engineering module. All of these communications are achieved through the

use of SNMP as defined in RFC 1157 [83].

SNMP is a part of the internet protocol suite, the standard which has been developed by the Internet Engineering Task Force (IETF). SNMP is classified as an application layer protocol, which sits at Layer 7 in the OSI Networking model. It is used by network management systems for monitoring devices attached to a particular network and carrying out administrative tasks.

In this work, the SNMP module, which is used in OptiFlow, must have Windows support since OptiFlow is a Microsoft Windows application. WinSNMP [84] provides a programming interface for network management applications running on Microsoft Windows as well as Linux platforms, thus enabling OptiFlow to make use of high level SNMP function calls.

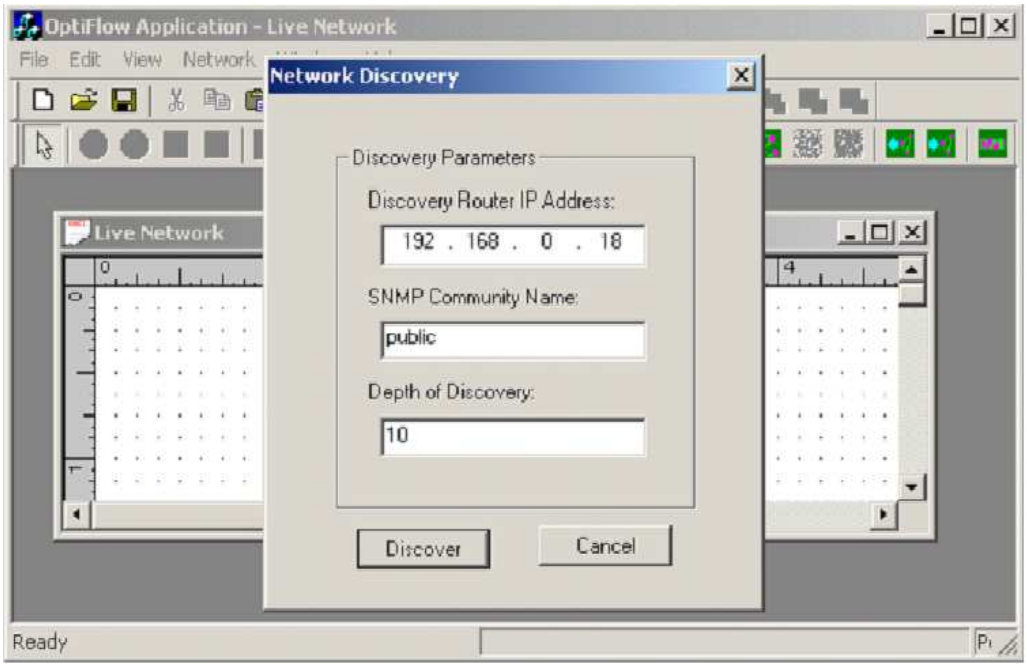

Figure 7.3: Network Discovery

A significant function carried out by the network interface module is network discovery. The purpose of network discovery is to discover the network topology under management consideration. This is useful if the network op-

erator is not quite sure about the network topology. It is certainly true that a network manager would be expected to have a detailed network map consisting of all of the network elements. However, due to network dynamics, some network components might be experiencing an outage, leaving the network map obsolete. Hence, the network discovery functionality is an important functionality for the tool as it can alert the manager to components that may not be operating correctly.

The discovery mechanism uses a depth first search recursion, where the user determines the depth of the search (see Figure 7.3). The consequence of having a small depth is that it may not be possible to explore and map the entire network. However, setting this depth to a large value will consume more time in exploring the network. In terms of the software architecture, network discovery is one of the tasks that the Network Interface must perform to populate the Network Topology data structure. One particular function that is required from the Network Interface is getting a list of OSPF adjacent hosts for a device. From the list of adjacent hosts, we are able to discover the network topology provided that the routers are running an OSPF routing process.

7.2.2 OptiFlow's Engines

OptiFlow's engines consist of five sub-modules. They are:

- *Network Topology* is a data container that holds information collected from the network via the network interface module. This is a collection of network objects which represent the data from a live network, such as links, routers, OD-Pairs, paths, etc.

- *Link Load Measurement* collects the real time statistics of the load from

every link in the network. This measurement data is then fed to the traffic matrix inference module. Since IP traffic is bursty, instantaneous link load measurements might be useless. Hence, an averaging or smoothing process is required.

- *Traffic Matrix Inference* tries to estimate a traffic matrix in a network based on link load measurement. The inference process in this system might not be required if direct measurement of OD-Pair data is available. However, direct measurement requires some additional infrastructure and consumes significant router CPU time - thus making it less preferable.

- *Optimisation* uses information from the network topology and the traffic matrix estimation system to find a set of link weights according to the optimisation objective.

- *Weight Updates* determines which link weights need to be updated according to the optimisation module. It then devises appropriate SNMP messages to be sent to the network interface and thence to the real network.

To determine the link load (utilisation), OptiFlow needs to interrogate every router in the network about the number of bytes that have been sent via the router interfaces. This is done once in every specified period, which can be configured by the user. A byte count on the router interface is essentially the number of bytes that have been transferred on a uni-directional link. This byte count can easily be obtained by querying the router MIB (Management Information Base) database using SNMP messages. The instantaneous utilisation of this link can be defined as $\frac{\Delta_{\text{bytes count}}}{\text{measurement interval}}$. However, due to IP traffic

burstiness, one might be required to do averaging between past measurement results. An averaging value for the link utilisation at time t can be defined as follows:

$$\rho_t = \alpha\rho_{t-1} + (1-\alpha)\frac{\Delta_{\text{bytes count}}}{\text{measurement interval}}$$

where α is a weight factor that determines the importance of the previous measurement. This method of averaging is also known as exponential moving averaging, where the historical measurement data is taken into consideration for the current measurement. The α value that we use ranged from 0.1 to 0.3.

One of the input requirements for the optimisation module is the set of origin-destination pair traffic estimates. This can be obtained by direct measurement or through inference techniques. Direct measurement can be achieved through the use of Netflow – which is a proprietary product of Cisco. Netflow can be installed on Cisco routers and it collects per-flow basis statistics. However, Netflow can only be installed on high-end routers because it uses a router's CPU time extensively.

OptiFlow implements traffic matrix estimation based on link load measurements. It is noted that inference techniques have gained popularity over direct measurement techniques because they are cheaper and they do not drain the router's CPU time. Several different inference techniques have been implemented in OptiFlow based on [85], [86] and [87].

The optimisation module is responsible for finding out the set of link weights for OSPF routing based on topology and traffic matrix information. It models intra-domain routing as shortest path forwarding based on the link weights. Aggregate traffic is assumed to be split equally if multiple paths exist to the destination. Based on this routing model, the resulting load on every link in the network can be calculated. The result of predicted link loads after

changes are carried out is presented to the user to make an informed decision as to whether to approve or reject the configuration changes. Currently, OptiFlow uses either CPLEX [67] or the glpk [75] LP solver to solve the LP optimisation problem. However, other methods or heuristics are possible, such as Lagrangian relaxation or Tabu search to solve the weight set optimisation problem.

When the optimisation is completed, some of the metrics in the network might need to be changed. The weight updates module takes the information from the optimisation module about the metrics on links that should be changed including their new metric values. It then devises appropriate messages for the SNMP Interface which are then translated into SNMP messages for routers to carry out configuration changes.

7.2.3 Steering

The Human Control Interface module is one of the most important modules in OptiFlow. It handles communication between OptiFlow and its users. Acting as an abstraction layer between the topology and network data to the OptiFlow user, this module provides a graphical display of the OSPF network. The topology is drawn on the canvas in the application.

Link congestion detection analyses data from the link load measurements. Depending on user settings, this module can automatically trigger the optimisation module and execute weight changes or prompt the user when a link utilisation exceeds a user-configurable threshold.

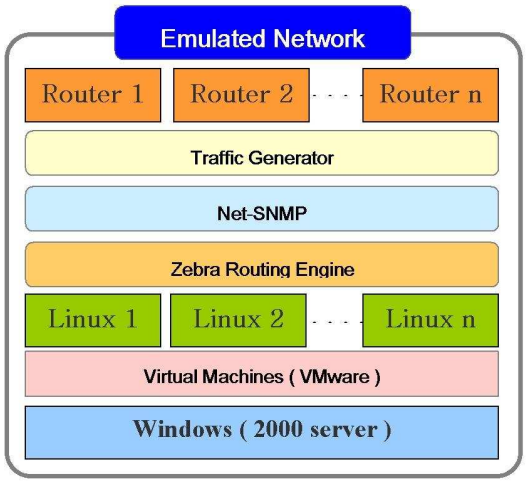

Figure 7.4: Virtual network test-bed structure

7.3 Virtual Network Building Blocks

Ideally, the OptifFlow application should have been tested on a fully operational IP network of reasonable size. However, due to a lack of resources and access to a significant sized network, a virtual network test-bed has been used. A virtual network test-bed consists of a collection of interconnected virtual machines that each run a separate OSPF routing process.

VMWare

VMWare [88] is a software application that provides computer virtualisation on a standard PC (with a generic Intel architecture). It allows users to set up multiple virtual computers and to use one or more of these virtual machines simultaneously. A virtual machine can have its own operating system. A virtual machine can also be configured to have components such as a physical PC, hard disks and network interfaces.

VMWare also provides the networking functionality framework to provide Ethernet bus interconnects among virtual machines. A point to point connec-

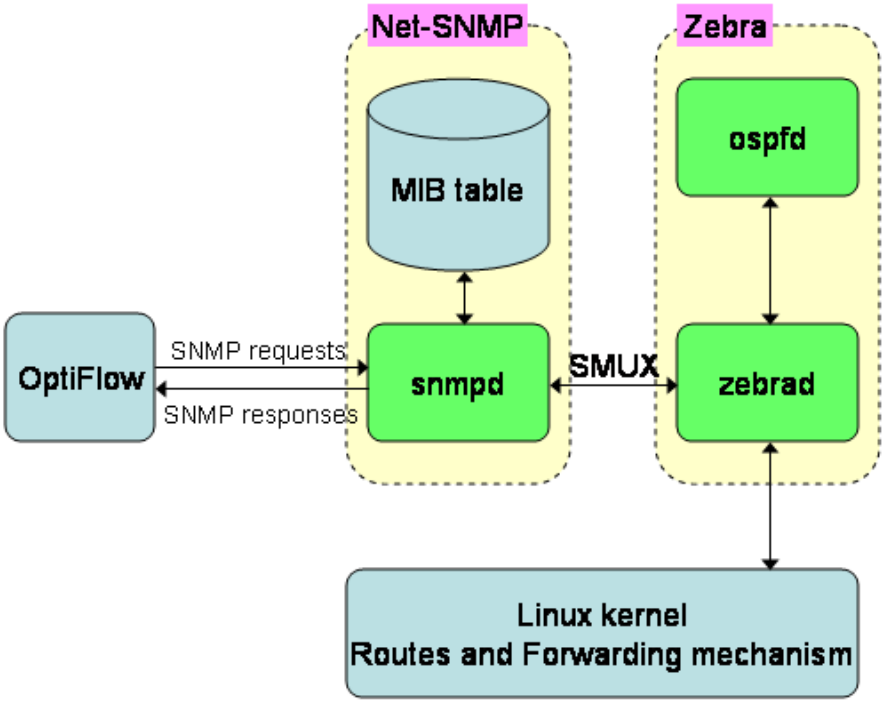

Figure 7.5: A Virtual Router structure

tion between two virtual machines can be realised using a single Ethernet bus shared between them. However, the number of Ethernet buses is currently limited to nine. Although, this feature is not properly documented in the manual, we have found out this feature through our experiments.

Zebra

Zebra [79] is a software routing package that provides a TCP/IP based routing service with routing protocol support. This software package runs under the Linux/BSD operating system. Zebra allows a PC with the Linux/BSD operating system to become a router. In other words, a PC with Zebra installed acts like a dedicated router. Zebra supports multiple routing protocols such as OSPF, RIP and BGP allowing PCs to exchange routing information with others in large and dynamic virtual networks.

Zebra provides a mechanism for setting up an interface, adding static routes, viewing forwarding tables, etc. It is an abstraction layer of Linux based router configurations, which typically are done by executing ifconfig, route and netstat commands. Using the command line interface provided, one can change the configuration and view routing table information from the Zebra terminal.

Zebra operates on the control plane of the router model. It communicates with other routers to build a global view of the network using routing protocols. Having completed the process, Zebra installs a suitable route into its routing table. The operation of the out-going interface lookup and forwarding of incoming packets is done by the Linux kernel.

`ospfd` is an OSPF daemon for the routing process and distributed as part of Zebra. It implements the OSPFv2 standard. It maintains OSPF database and exchanges OSPF messages. It injects the routers discovered by ospfd process based on OSPF messages to zebrad. To the best of our knowledge, Zebra provides the closest basic functionality for the OSPF routing protocol when compared to OSPF functionality in a Cisco router. At present, Zebra is no longer being developed. As an alternative, a fork of Zebra, called Quagga, is under development.

Net-SNMP

By default, the information of the machine in Linux OS is collected under the */proc* file system. On the other hand, a router usually maintains its information in a table called the Management Information Base (MIB) table. MIB is defined as a type of database used to manage the devices in a communications network. It comprises a collection of objects in a (virtual) database used

to manage entities (such as routers and switches) in a network. Net-SNMP also provides an implementation of the MIB table, thus allowing the snmpd daemon to extract information from MIB table.

To make a router respond to SNMP requests sent by OptiFlow, the router needs to have a daemon that listens for such requests. The daemon, called `snmpd`, is part of Net-SNMP distribution. The requests can be requests for information or for setting up a value in the MIB table.

Zebra itself does not support SNMP agent functionality. But in conjunction with an SNMP agent (in this case its `snmpd` daemon), Zebra provides routing protocol MIBs. The communication between `snmpd` and `zebrad` is governed by SNMP Multiplexing (SMUX). SMUX resolves the problem of information exchanges between running processes (in our case, they are `zebrad` and `snmpd`).

7.4 Experimental Setup and Preliminary Results

This section describes experimental setup and results during our experiments on a physical network and on a virtual network. It also outlines the issues that we faced during our experiments.

7.4.1 Network Discovery of a Physical Network

OptiFlow has been tested on an actual physical network in the RMIT University Router Laboratory facility (its topology is given in Figure 7.6). In this laboratory, there are six routers running the OSPF routing protocol. These six routers are connected by a serial PPP line, which runs either at 1.544 or 2 Mbps. Connected to the last two routers are Cisco Catalyst switches, that

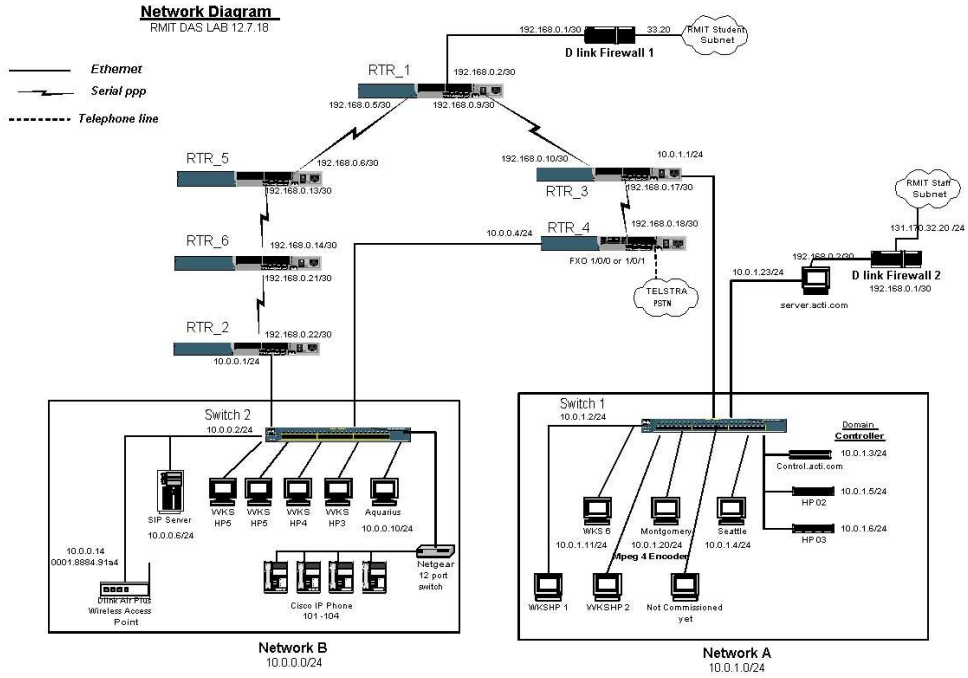

Figure 7.6: RMIT Router Laboratory

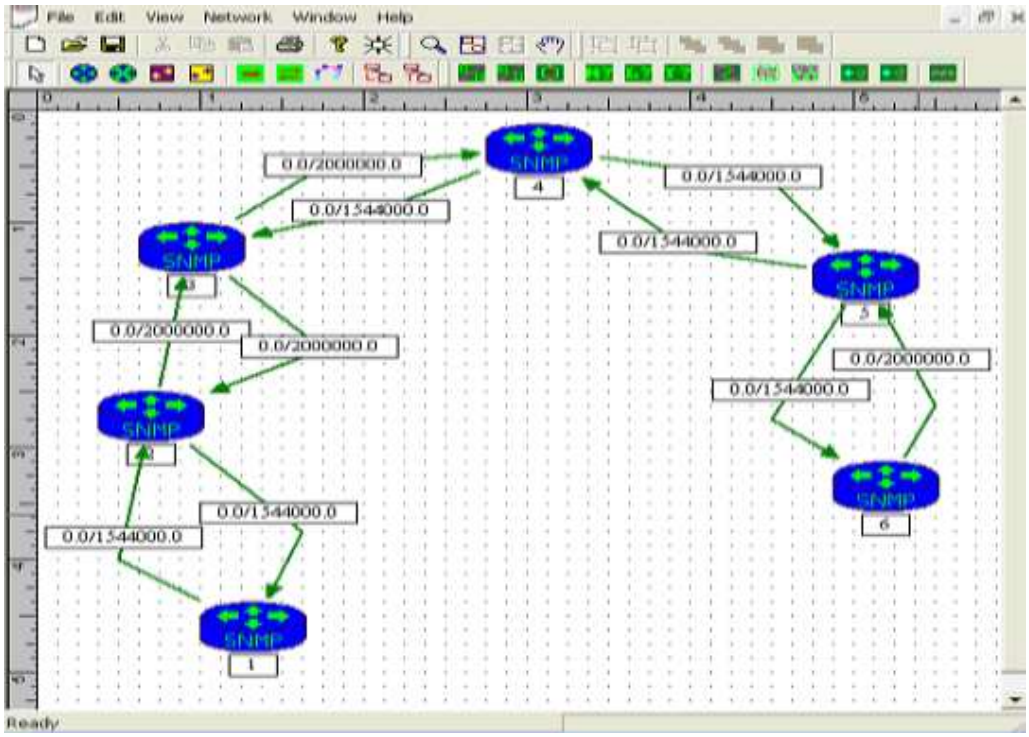

Figure 7.7: OptiFlow topology on RMIT Router Laboratory

provide Ethernet connections to the workstation computers. The result of network discovery is given in Figure 7.7.

In this setup, OptiFlow managed to discover all OSPF routers in the network. OptiFlow does not include the switches because they are not running the OSPF routing process. The switches are just the end points that connect to workstation computers. OptiFlow also managed to gather the properties of the router such as its name, link connectivity, up-time, etc.

7.4.2 Experiments on a Virtual Network

Due to hardware and access limitations, we were unable to alter the network topology in this setup. Hence, to carry out further tests for OptiFlow, we opted to build virtual networks based on the components described in the previous section, namely VMWare, which provides virtual machine support, and the Zebra routing engine, which maintains routing and connectivity. The idea behind it is to run several interconnected virtual machines with IP routing enabled (OSPF in our case).

We built the well-known "trap topology" configuration into the virtual network test-bed. This topology has 5 routers and they are connected with 6 uni-directional links. The speed of these links has been limited to 5 Mbps. There are two OD-pairs, namely node 1 to node 4 and node 2 to node 4. We used a customised Windows application to generate traffic for these two OD-pairs. The traffic generator enables us to send packets to a destination by entering the destination's IP address. The first OD-pair is 3 Mbps and the second is 6 Mbps.

The shortest path routing will cause congestion on the link between node 2 and 4. The congestion is detected by OptiFlow and it is indicated as a red

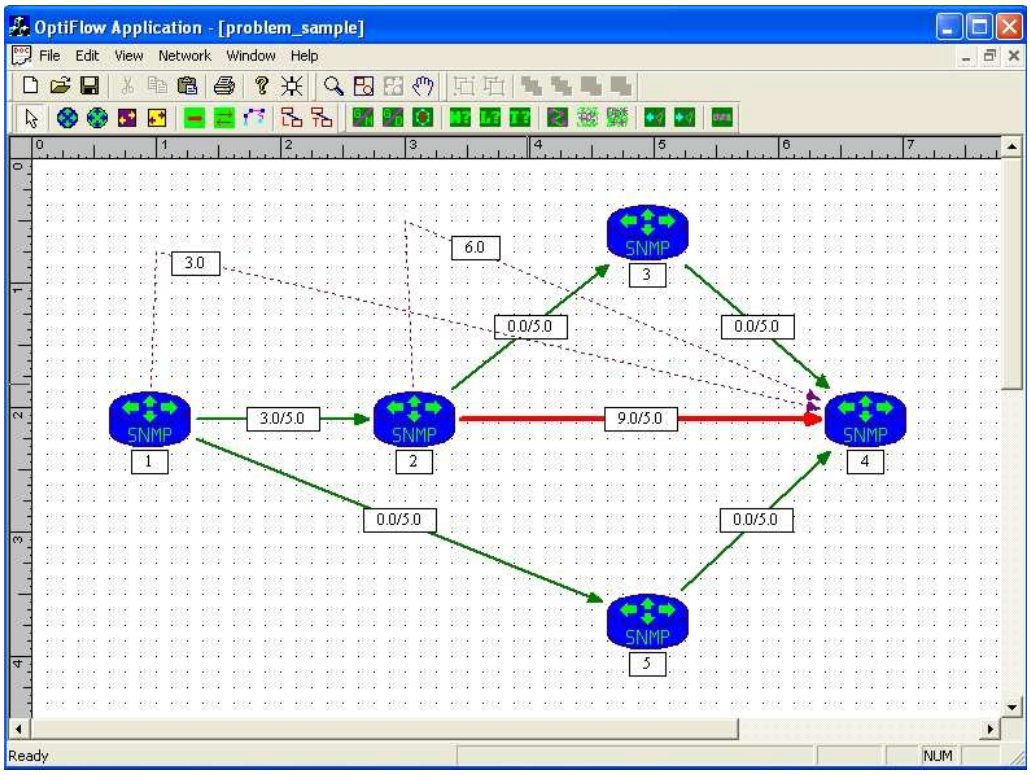

Figure 7.8: Network under observation

coloured link in the canvas of the tool. The screen-shot of OptiFlow whilst detecting congestion is given in 7.8. The pairs of numbers on the links denote the amount of traffic on the link and the link capacity.

The OptiFlow monitoring module starts up upon detecting congestion and prompts the user as to whether an optimisation process should be invoked. A series of processes will be started, if the user chooses to optimise the network. The processes are:

- A link load measurement is required to calculate a new traffic matrix.
- A traffic matrix estimation process provides OD-Pair information for the optimisation step.
- LP problem formulation and solution to find out the set of new link

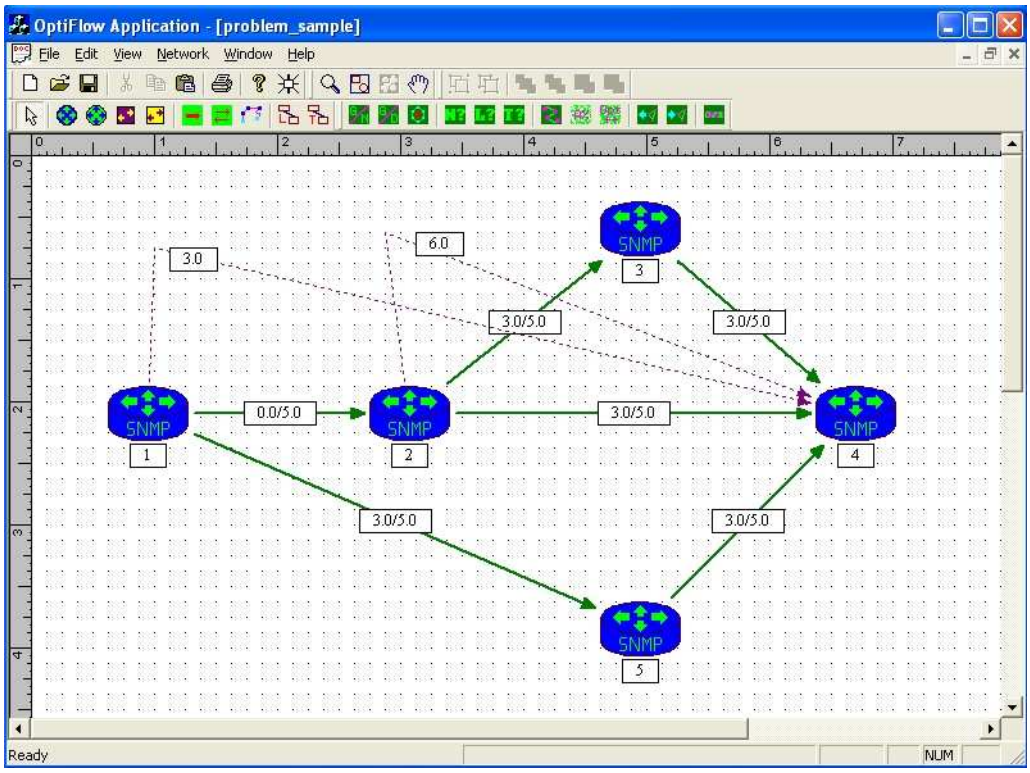

Figure 7.9: Relieving network congestion by changing weights

weights.

Once the new set of weights is distributed (in this case the congested link weight is updated), routers recompute their shortest paths and forward traffic based on the new shortest paths. In this experiment, the shortest path from node 1 to node 4 is via node 5. Node 2 will have two equal length paths to reach node 4. By enabling load balancing mechanism on node 2 in the Linux kernel, we managed to split traffic between the direct link and node 3; hence removing congestion as depicted in Figure 7.9.

7.4.3 Experimental Issues

In our experiments, the traffic generator provides a constant traffic stream encapsulated in constant-sized UDP packets. However, in real networks, a

link is likely to carry hundreds or even thousands of flows with different sized packets. A perfect 50%-50% splitting as given above rarely occurs in practice, because of the different sizes of flows. Furthermore, a splitting process is done based on a destination IP address and a pseudo-random generator.

A weight adjustment process by using SNMP protocol is not trivial. Zebra routing software needs to be patched to allow weight changes via SNMP. However, due to the use of a proprietary IOS, it is impossible to alter the code in Cisco routers. One of the possible causes is a compatibility issue related to the message interface and the SNMP process that is running on the router. Cisco IOS seems to prevent modification of the link metric entry (maybe due to security reasons and a compatibility issue across different versions of SNMP). Regardless, weight changes can still be achieved either automatically opening a telnet session to the router or doing it manually.

We have shown that it is possible to emulate a network within a single PC. However, it comes at the cost of poor response time, differences between the real routing process, unreliable emulated interfaces, etc. One cannot rely on the software emulated interface because the interface speed might be dependent on the load that the host machine has. The validity of the traffic generator needs to be improved.

As part of ongoing work, we are evolving the tool to operate reliably in overloaded networks. At the moment, the keep-alive detection mechanism to monitor the routers is done through SNMP queries. However, as the network is overloaded, this mechanism is no longer reliable because SNMP queries are congested and lost. A more elegant solution involves the monitoring of OSPF messages in the network that indicates the routers' status. This has an added advantage, which is suppressing the amount of traffic due to SNMP messages.

7.5 Conclusions

OptiFlow integrates the functionality of network monitoring OSPF networks. We have implemented OptiFlow as a prototype of a network management tool. It also allows the user to try out different routing strategies in a enclosed environment, away from the real network. A flexible interface allows the user to manipulate the configuration of the routers. A weight setting based optimisation for OSPF networks has been implemented in OptiFlow as well.

Chapter 8

Conclusions

The main objective of this thesis has been to study traffic engineering methods in the current IP based networks. Current techniques based on heuristics have been considered too slow for online traffic engineering applications. However, LP-based techniques are not yet recognised as a candidate for traffic engineering mainly due to the uniqueness of the solution. We show that it is possible to estimate how good the solution is by counting the number of splitting ratios created, when LP solution is used to modify the weight and route traffic. To address the issue with the amount of time it takes for an OSPF network to converge after a weight setting is invoked, we carried out simulation study to determine network convergence time. We found that the component major components of the convergence time are the link propagation delay and timers associated with the OSPF processes to dampen network instability .

In MPLS networks, service providers can utilise the MPLS explicit route feature to pin down routes. This is often done since the route is preferred to the others. In the problem that we looked at, given a bandwidth requirement of OD-Pairs in the network, we formulate path selection mechanism with different objective functions (minimising utilisation / network cost) and different

network constraints (single path / multi path routing).

During transition of IP networks to MPLS networks, service providers still prefer to route non-premium traffic using the underlying IGP mechanisms. The use of MPLS is just limited to premium traffic. In this work it is shown that it is possible to obtain routing patterns for both types of traffic. Furthermore, the solution is readily implemented using a set of weights as IGP metrics and a set of MPLS explicit routes.

How much improvement has MPLS technology brought to the networking industries? To answer this, we carried out comparison study of different routing strategies, namely, default OSPF routing using Cisco metric, OSPF routing optimised using LP-based weight setting and MPLS explicit routing using Marginal Increase Heuristic. Although, MPLS has been shown as a clear winner of all, since it can deliver 100%, given the same underlying network topology, all the routers in the domain must be MPLS capable. Unfortunately, the cost factor might hinder small / medium scale network carriers to upgrade their networks to MPLS networks.

Realising that network industry only have limited tools to carry out network management functions, in this thesis, we outlined the design and implementation of OptiFlow, network capacity management tool in shortest path based networks. The tool includes traffic matrix estimation and traffic engineering functionality. The tool is a collaboration work with Mr. Suyong Eum [86] [89] in the former Australian Telecommunication Cooperative Research Centre (ATcrc).

8.1 Summary of Contributions

A final list of the contributions of this thesis is given below:

- Development of dual simplex method that allows fewer iterations in comparison to the simplex method for solving the multi-commodity flow problem with a fewer number of iterations.

- Introduction of the idea of splitting ratio that measures the degree of uniqueness of the routing solution. This allows us to estimate how far the solution from the LP optimum solution.

- Study of network routing speed of convergence due to the weight changes in OPSF networks. We observed how long does it take for the weight updates propagate throughout the network using ns-2 simulator.

- Development of two mathematical models that allow explicit route configuration in MPLS networks (with minimal re-configurations)

- Development and numerical study of an MIP/LP-based optimisation model that takes into account multiple traffic streams in IP networks. The traffic streams, depending on QoS or bandwidth requirements, are routed using different technologies, i.e. native IP routing or MPLS.

- Comparison study of two different traffic engineering methods to optimise OSPF and MPLS networks against the default OSPF routing recommended by Cisco. The comparison study uses ns-2 application to investigate packet loss in small to medium sized networks given multiple demand matrix.

- Design and implementation of network management software, OptiFlow, a Microsoft Windows application that implements network management functionalities, including traffic matrix estimation and network congestion removal.

- Design and implementation of a virtual network test bed using VMWare, Zebra and Net-SNMP in a single PC to emulate a small network.

8.2 Future Research

One of major issues with traffic engineering methods is that their scope is just limited to domains that are under the service providers' control. Customers require end to end QoS, hence traffic engineering approaches need to address the optimisation of other service providers' domains with routers running different routing protocols. Although there is a system to deliver end to end QoS, this would require a huge number of service providers to agree to deploying the same the mechanisms simultaneously. A complex monitoring and payment mechanism between all of the different service providers will also arise as a consequence.

We believe that inter-domain QoS requires a fair amount of research due to the inflexibility of the de-facto inter-domain routing protocol known as the Border Gateway Protocol (BGP-4), which works based on a set of policies rather than mtrics used in shortest path computations. The interaction between intra-domain and inter-domain routing needs to be considered as well; as we carried out intra or inter-domain traffic engineering.

Capacity expansion also needs to be carried out carefully. If some link capacity is upgraded and the default routing metric, such as inverse of capacity, is used, then the upgraded link will attract more traffic. However, congestion might occur on the upstream side of this link due to additional traffic. Hence, capacity expansion planning is essential, otherwise one would just be shifting congestion from one part of the network to another. This is certainly true in road traffic and has already been identified as a problem, [90].

Throughout our modelling, we have been reliant on LP solvers to solve MILP by using a branch and bound technique. Although computing technology advancements and advances to computation techniques have contributed significantly to improvements in solution time, we believe that heuristics may be a better way to solve MILP type problems if the solution time is seen to be critical. However, the trade-off is that the solution given by the heuristic may not always be close to optimal. Development of heuristics tailored to our MILP model will result in a faster computation time.

The work in relation to calculation time can also be extended. In Chapter 3, we briefly stated the computation time for the network that we obtained from our experiments. However, this can be extended to estimate the calculation time for a network with an arbitrary number of nodes, links and OD-Pairs. We envisage that the LP problem size will be changing. However, the performance of LP solvers with different problem sizes is still not well-known.

Furthermore, link updates occur quite often during regular network operations. The source of link updates can be link failures or just link status fluctuations (regardless of whether this is a true or false notification) [52]. Although certain measures have been taken to dampen the frequency of link updates, a study about how the frequency of link updates affects the network dynamics is required. Network dynamic is such a complex issue because of the aspects involved. These include the underlying network architecture, the routing process used, the long and short term traffic patterns. The combination of these aspect makes the network dynamic a complicated yet interesting to explore further.

In this work, we stipulated that updates that can be done in order of a few seconds are close to optimal in comparison to other methods that produce solutions in order of minutes or hours. In reality, the optimal point is not

really known, because network managers have to consider network dynamic aspects as mentioned above. An optimal point from the theoretical perspective is useful, but it is better to discover the optimal point for network operators.

Bibliography

[1] Z. Wang, *Internet QoS: Architectures and Mechanisms for Quality of Service*, 1st ed. Morgan Kaufmann, 2001.

[2] P. Willis, *Carrier-scale IP Networks*. The Institution of Electrical Engineers, 2001.

[3] D. Awduche, J. Malcolm, J. Agogbua, M. O'Dell, and J. McManus, "Requirements for Traffic Engineering Over MPLS," 1999. [Online]. Available: http://www.ietf.org/rfc/rfc2702.txt

[4] D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao, "Overview and Principles of Internet Traffic Engineering," 2002. [Online]. Available: http://www.ietf.org/rfc/rfc3272.txt

[5] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, and J. Rexford, "NetScope: traffic engineering for IP networks," *Network, IEEE*, vol. 14, no. 2, pp. 11–19, 2000.

[6] B. Fortz and M. Thorup, "Internet traffic engineering by optimizing OSPF weights," in *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 2, 2000, pp. 519–528 vol.2.

[7] S. Iyer, S. Bhattacharyya, N. Taft, and C. Diot, "An approach to alleviate link overload asa observed on an IP backbone," in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies. IEEE*, vol. 1, 2003, pp. 406–416.

[8] E. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol Label Switching Architecture," 2001. [Online]. Available: http://www.ietf.org/rfc/rfc3031.txt

[9] X. Xiao, A. Hannan, B. Bailey, and L. Ni, "Traffic engineering with MPLS in the Internet," *Network, IEEE*, vol. 14, no. 2, pp. 28–33, 2000.

[10] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms and Applications*, 1st ed. Prentice Hall, 1993.

[11] M. Pioro and D. Medhi, *Routing, Flow and Capacity Design in Communication and Computer Networks*, 1st ed. Morgan Kaufmann, 2004.

[12] Y. Wang, Z. Wang, and L. Zhang, "Internet traffic engineering without full mesh overlaying," in *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 1, 2001, pp. 565–571.

[13] B. Gavish and I. Neuman, "A System for Routing and Capacity Assignment in Computer Communication Networks," *IEEE Transactions on Communications.*, vol. 37, no. 4, April 1989.

[14] Y. Wang and Z. Wang, "Explicit Routing Algorithms for Internet Traffic Engineering," in *Computer Communications and Networks, 1999. Proceedings. Eighth International Conference on*, 2001, pp. 582–588.

[15] J. E. Burns, T. J. Ott, A. E. Krzesinski, and K. E. Muller, "Path selection and bandwidth allocation in MPLS networks," *Performance Evaluation*, vol. 52, pp. 133–152, 2004.

[16] J. M. Kleinberg, "Single-source unsplittable flow," in *FOCS '96: Proceedings of the 37th Annual Symposium on Foundations of Computer Science*. Washington, DC, USA: IEEE Computer Society, 1996, p. 68.

[17] R. Guerin, H. Ahmadi, and M. Naghshineh, "Equivalent capacity and its application to bandwidth allocation in high-speed networks," *Selected Areas in Communications, IEEE Journal on*, vol. 9, no. 7, pp. 968–981, 1991.

[18] I. Atov, "Design of IP Networks with End-to-End Performance Guarantees," Ph.D. dissertation, RMIT University, Melbourne, Australia, 2003.

[19] J. Murphy, R. Suryasaputra, W. Lloyd-Smith, X. V. Doan, R. Nelson, and R. Harris, "Dynamic Resource Management Using LP Weight Setting in an MPLS Domain," in *Next Generation Teletraffic and Wired/Wireless Advanced Networking (NEW2AN'04)*, 2004.

[20] R. Suryasaputra, J. Murphy, and R. Harris, "Dousing Hot Spots in OSPF Networks," in *ATNAC 2004*, Sydney, Australia, December 2004.

[21] B. Lloyd-Smith, R. Suryasputra, J. Murphy, and R. Harris, "Removing hot spots using a simple LP method and why it works," in *ATNAC*, Melbourne, 2003.

[22] J. Murphy, R. Suryasaputra, and R. Harris, "Link Congestion Avoidance Using Weight Setting in an MPLS Network - a Simple LP Approach," in *ATNAC 2003*, Melbourne, Australia, December 2003.

[23] R. Suryasaputra, A. A. Kist, and R. J. Harris, "Verification of MPLS Traffic Engineering Techniques," in *ICON2005. International Conference on Networks*, 2005.

[24] R. Suryasaputra, A. A. Kist, H. L. Ferra, R. A. Palmer, M. J. Dale, and R. J. Harris, "Comparison of Intra-Domain Traffic Engineering Methods," in *World Teletraffic Congress WTC*, 2006.

[25] R. Suryasaputra, A. A. Kist, and R. J. Harris, "OptiFlow - A Capacity Management Tool," in *TENCON 2005. 2005 IEEE Region 10 Conference. IEEE*, 2005.

[26] ——, "Reconfiguration in Multi Class of Services Networks," in *ATNAC*, Melbourne, 2006.

[27] C. Huitema, *Routing in the Internet*, 2nd ed. Prentice Hall, 2000.

[28] B. Fortz and M. Thorup, "Optimizing OSPF/IS-IS weights in a changing world," *Selected Areas in Communications, IEEE Journal on*, vol. 20, no. 4, pp. 756–767, 2002.

[29] S. Srivastava, G. Agrawal, M. Pióro, and D. Medhi, "Determining Feasible Weight System under Various Objectives for OSPF Networks," University of Missouri-Kansas City, Tech. Rep., September 2003.

[30] A. Jüttner, B. Szviatovski, I. Mécs, and Z. Rajkó, "Lagrange relaxation based method for the qos routing problem," in *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 2, 2001, pp. 859–868.

[31] E. Mulyana and U. Killat, "An Alternative Genetic Algortihm to Optimise OSPF Weights," in *15th ITC Specialist Seminar. Internet Traffic Engineering and Traffic Management*, 2002.

[32] M. Ericsson, M. Resende, and P. Pardalos, "A Genetic Algorithm for the Weight Setting Problem in OSPF Routing," *Journal of Combinatorial Optimization*, vol. 6, pp. 299–333, 2002.

[33] P. Gajowniczek, M. Pióro, A. Szentesi, and J. Harmatos, "Solving an OSPF Routing Problem with Simulated Allocation," in *FOCS '96: Proceedings of the 37th Annual Symposium on Foundations of Computer Science*, 2000.

[34] A. Bley, M. Grotchel, and R. Wessaly, "Design of broadband virtual private networks: Model and heuristics for the B-WiN," Tech. Rep. Technical Report SC 98-13, KonradZuse -Zentrum fur Informationstechnik Berlin. To appear in Proc. DIMACS Workshop on Robust Communication Networks and Survivability, AMS-DIMACS Series., 1998.

[35] M. Pióro, A. Szentesi, J. Harmatos, A. Jüttner, P. Gajowniczek, and S. Kozdrowski, "On Open Shortest Path First related Network Optimisation Problems," *Performance Evaluation*, vol. 48, pp. 201–223, 2002.

[36] A. Sridharan, R. Guerin, and C. Diot, "Achieving near-optimal traffic engineering solutions for current OSPF/IS-IS networks," in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies. IEEE*, vol. 2, 2003, pp. 1167–1177.

[37] M. Thorup and M. Roughan, "Avoiding Ties in Shortest Path Routing." [Online]. Available: http://www.research.att.com/~mthorup/PAPERS/ties_ospf.ps

[38] D. O. Awduche, "MPLS and Traffic Engineering in IP Networks," *Communications Magazine, IEEE*, pp. 42–47, 1999.

[39] A. Elwalid, C. Jin, S. H. Low, and I. Widjaja, "MATE: MPLS Adaptive Traffic Engineering," in *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, 2001, pp. 1300–1309.

[40] M. S. Kodialam and T. V. Lakshman, "Minimum Interference Routing with Applications to MPLS Traffic Engineering," in *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, 2000, pp. 884–893.

[41] A. Bagula, M. Botha, and A. Krzesinski, "Online traffic engineering: the least interference optimization algorithm," in *Communications, 2004. ICC '04. IEEE International Conference on*, vol. 2, 2004, pp. 1232–1236.

[42] S. Köhler and A. Binzenhfer, "MPLS Traffic Engineering in OSPF Networks - A combined approach," in *International Teletraffic Congress ITC18*, 2003.

[43] W. Ben Ameur, N. Michel, E. Gourdin, and B. Liau, "Routing strategies for IP networks," *Telektronikk*, vol. 2/3, pp. 145–158, 2001.

[44] J. Garcia, A. Rachdi, and O. Brun, "Optimal LSP Placement with QoS Constraints in DiffServ/MPLS Networks," in *International Teletraffic Congress ITC18*, 2003.

[45] A. Riedl, "Routing Optimization and Capacity Assignment in Multi-Service IP Networks," Ph.D. dissertation, Technische Universit$\ddot{a}$t München, Munchen, Germany, 2003.

[46] ——, "Optimized routing adaptation in IP networks utilizing OSPF and MPLS," in *Communications, 2003. ICC '03. IEEE International Conference on*, vol. 3, 2003, pp. 1754–1758.

[47] E. Mulyana and U. Killat, "An Offline Hybrid IGP/MPLS Traffic Engineering Approach under LSP Constraints," in *INOC2003. International Network Optimization Conference.*, W. Ben-Ameur and A. Petrowski, Eds., 2003.

[48] ——, "Optimization of IP Networks in Various Hybrid IGP/MPLS Routing Schemes," in *12th Conference on Measuring, Modelling and Evaluation of Computer and Communication Systems*, 2004.

[49] P. Trimintzios, L. Georgiadis, G. Pavlou, D. Griffin, C. Cavalcanti, P. Georgatsos, and C. Jacquenet, "Engineering the Multi-Service Internet: MPLS and IP-based Techniques," in *ICT'2001, IEEE International Conference on Telecommunications. Proceedings. IEEE*, vol. 3, 2001, pp. 129–134.

[50] "ns-2 Simulator." [Online]. Available: http://www.isi.edu/ns/nsnam

[51] A. Fumagalli and L. Valcarenghi, "IP restoration vs. WDM protection: is there an optimal choice?" *Network, IEEE*, vol. 14, no. 6, pp. 34–41, 2000.

[52] G. Iannaccone, C. N. Chuah, R. Mortier, S. Bhattacharyya, and C. Diot, "Analysis of link failures in an IP backbone," in *Second ACM SIGCOMM Workshop on Internet measurement*, 2002.

[53] A. Nucci, B. Schroeder, S. Bhattacharyya, N. Taft, and C. Diot, "IGP Link Weight Assignment for Transient Link Failures," in *International Teletraffic Congress ITC18*, 2003.

[54] B. Fortz and M. Thorup, "Robust Optimization of OSPF/IS-IS Weights," in *INOC2003. International Network Optimization Conference.*, W. Ben-Ameur and A. Petrowski, Eds., 2003, pp. 225–230.

[55] J. Murphy, R. Harris, and R. Nelson, "Traffic Engineering Using OSPF Weights and Splitting Ratios," in *IFIP Conference*, Perth, Australia, 2003, pp. 277–287.

[56] H. Taha, *Operational Research: An Introduction*, 6th ed. New Jersey: Prentice Hall International, 1997.

[57] J. Murphy, X. V. Doan, R. Nelson, and R. Harris, "Capacity Management: Using The Dual Solution of the Multi-Commodity Flow Problem to Set OSPF Weights - a Fast Heuristic," ATcrc, Tech. Rep., January 2003.

[58] C. Alaettinoglu, V. Jacobson, and H. Yu, "Towards Milli-Second IGP Convergence (Internet Draft)," 2000. [Online]. Available: http://www.packetdesign.com/news/industry-publications/drafts/convergence.pdf

[59] K. Svensson, "Implementation of a simulation platform for IS-IS and simulation of Telia's IP network," Ph.D. dissertation, Chalmers University of Technology, Sweden, 2001.

[60] W. Ben-Ameur, E. Gourdin, and N. Bourquia, "Optimal routing for efficient Internet networks," in *Proceedings of the ECUMN 2002*, Colmar, France, 2002.

[61] S. Soh, L. Hiryanto, R. P. Gopalan, and S. Rai, "Dynamic Router Tables for Full Expansion/Compression IP Lookup," in *TENCON 2005. 2005 IEEE Region 10 Conference. IEEE*, 2005.

[62] I. Chaieb, J. Le Roux, and B. Cousin, "MPLS-TE Routing: Adopting a Generic Architecture and Evaluating Various Implementation Approaches," in *ATNAC*, Melbourne, 2006.

[63] A. Capone and F. Martignon, "Analysis of dynamic QoS routing algorithms for MPLS networks," in *Communications, 2004. ICC '04. IEEE International Conference on*, vol. 2, 2004, pp. 1192–1196.

[64] B. Wang, X. Su, and C. L. P. Chen, "A New Bandwidth Guaranteed Routing Algorithm for MPLS Traffic Engineering," in *Communications, 2002. ICC '02. IEEE International Conference on*, 2002, pp. 1001–1005.

[65] G. Figueiredo, N. da Fonseca, and J. Monteiro, "A minimum interference routing algorithm," in *Communications, 2004. ICC '04. IEEE International Conference on*, vol. 4, 2004, pp. 1942–1947.

[66] J. Y. Yen, "Finding the k shortest loopless paths in a network," *Management Science*, vol. 17, no. 11, pp. 712–716, 1971.

[67] "CPLEX LP Solver Version 7.1." [Online]. Available: http://www.ilog.com

[68] K. Calvert, M. Doar, and E. W. Zegura, "Modeling internet topology," *Communications Magazine, IEEE*, vol. 35, no. 6, pp. 160–163, 1997.

[69] K. Nichols and B. Carpenter, "Definition of Differentiated Services Per Domain Behaviors and Rules for their Specification," 2001. [Online]. Available: http://www.ietf.org/rfc/rfc3086.txt

[70] F. L. Faucheur and W. Lai, "Requirements for Support of Differentiated Services-aware MPLS Traffic Engineering," 2003. [Online]. Available: http://www.ietf.org/rfc/rfc3564.txt

[71] E. Horlait and N. Rouhana, "Differentiated Services and Integrated Services Use of MPLS," in *ISCC 2000. Fifth IEEE Symposium on Computers and Communications. Proceedings. IEEE*, 2000.

[72] T. Bonald, A. Proutiere, and J. Roberts, "Statistical Performance Guarantees for Streaming Flows using Expedited Forwarding," in *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 2, 2001, pp. 1104–1112 vol.2.

[73] F. Tommasi, S. Molendini, and A. Tricoo, "Mapping of IntServ/RSVP Reservations into MPLS domains," in *SOFTCOM2002. International Conference on Software, Telecommunications and Computer Networks. Proceedings. IEEE*, 2002.

[74] R. J. Harris, "A Mathematical Programming Model for Service Protection in a Telecommunications Network," in *International Teletraffic Congress ITC13*, 1991.

[75] "GNU Linear Programming Kit 4.4," 2004. [Online]. Available: ftp://ftp.gnu.org/gnu/glpk

[76] M. J. Dale, H. L. Ferra, and R. A. Palmer, "Fast MPLS Network Optimisation using Machine Learning," in *TENCON 2005. 2005 IEEE Region 10 Conference. IEEE*, vol. 1, 2005, pp. 971–976.

[77] F. Poppe, S. Van den Bosch, P. de La Vallée-Poussin, H. Van Hove, and G. Petit, "Choosing the Objectives for Traffic Engineering in IP Backbone Networks Based on Quality-of-Service Requirements," in *QOFIS2000*, J. Crowcroft, J. Roberts, and M. MSmirnov, Eds., vol. 1. Berlin: Springer-Verlag, 2000, pp. 129–140.

[78] Y. Zhang, M. Roughan, N. Duffield, and A. Greenberg, "Fast accurate computation of large-scale IP traffic matrices from link loads," in *Proceedings of the 2003 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*. San Diego, CA, USA: ACM Press, 2003, pp. 206–217.

[79] K. Ishiguro, "GNU Zebra," 2003. [Online]. Available: GNU http://www.zebra.org

[80] F. Baker and R. Coltun, "OSPF Version 2 Management Information Base," 1995. [Online]. Available: http://www.ietf.org/rfc/rfc1850.txt

[81] O. Goldschmidt, "ISP Backbone Traffic Inference Methods to Support Traffic Engineering," in *In Internet Statistics and Metrics Analysis (ISMA) Workshop*, San Diego, CA, December 2000, pp. 1063–1075.

[82] A. Medina, N. Taft, K. Salamatian, S. Bhattacharyya, and C. Diot, "Traffic Matrix Estimation: Existing Techniques and New Directions," in *ACM SIGCOMM*, August 2002.

[83] J. J. Case, M. Fedor, M. Schoffstall, and J. Davin, "A Simple Network Management Protocol (SNMP)," 1990. [Online]. Available: http://www.ietf.org/rfc/rfc1157.txt

[84] "WinSNMP." [Online]. Available: http://www.winsnmp.org/

[85] S. Eum, J. Murphy, and R. Harris, "TomoKruithof vs Tomogravity for Backbone Networks," in *ATNAC*, Sydney, 2004.

[86] ——, "A Fast Accurate LP Approach for Traffic Matrix Estimation," in *International Teletraffic Congress ITC19*, Beijing, China, 2005.

[87] ——, "A Failure Analysis of the Tomogravity and EM Methods," in *TENCON*, December 2005.

[88] "VMWare Workstation 4.5." [Online]. Available: http://www.vmware.com/products/desktop/ws_features.html

[89] S. Eum, R. J. Harris, and I. Atov, "Traffic Matrix Estimation with Markov Arrival Process of the Order of Two (MAP-2)," in *International Teletraffic Congress ITC-20 - Under submission*, 2007.

[90] "The Road Network Paradox." [Online]. Available: http://www.davros.org/science/roadparadox.html

Abbreviations

AF	Assured Forwarding
AS	Autonomous System
ATM	Asynchronous Transfer Mode
BE	Best Effort
BGP	Border Gateway Protocol
CATT	Centre of Advanced Technology in Telecomunications
CAPEX	Capital Expenditure
CBR	Class Based Routing or Constant Bit Rate
DiffServ	Differentiated Service
ECMP	Equal Cost Multi Path
EF	Expedited Forwarding
EGP	Exterior Gateway Protocol
FR	Frame Relay
IDRP	Inter-Domain Routing Protocol
IGP	Interior Gateway Protocol
IGRP	Interior Gateway Routing Protocol
ILP	Integer Linear Programming
IntServ	Integrated Service
IP	Internet Protocol
IS-IS	Intermediate System-Intermediate System
ISP	Internet Service Provider
LP	Linear Programming
LSA	Link State Advertisement

LSP	Label Switched Path
LSP	Link State Packet
MIB	Management Information Base
MIH	Marginal Increase Heuristic
MILP	Mixed Integer Linear Programming
MPLS	Multi Protocol Label Switching
NMS	Network Management Systems
NN	Neural Networks
NP	Non-polynomial
OD	Origin Destination
OPEX	Operating Expenditure
OSPF	Open Shortest Path First
RIP	Routing Information Protocol
SLA	Service Level Agreement
SNMP	Simple Network Management Protocol
SPF	Shortest Path First
TE	Traffic Engineering
TOS	Type of Service
VC	Virtual Circuit

List of Figures

List of Tables

Appendix

Dual Simplex to Recover Feasibility

A standard LP problem can be defined as follows

$$\text{Maximise or minimise} \quad z = \sum_{j=1}^{n} c_j x_j \tag{8.1a}$$

subject to

$$\sum_{j=1}^{n} a_{ij} x_j = b_i, \quad i = 1, 2, \ldots, m \tag{8.1b}$$

$$x_j \geq 0, \quad j = 1, 2, \ldots, n \tag{8.1c}$$

The variables x_j, $j = 1, 2, \ldots, n$, include the surplus and slacks, if any, are to convert any inequality constraints to equality constraints.

The revised simplex method is presented in [56]. It is based exactly on the same steps used in the simplex method. The only difference is that, the basis matrix of the next iteration is computed based on the previous basis matrix and its inverse, whereas, row operations are needed in the original simplex method. The advantages of the revised simplex method are that there are a smaller number of arithmetic operations and smaller round-off errors during the process.

However, the revised simplex method cannot be directly used in our prob-

lem. The (revised) simplex method requires that the initial solution be feasible but not necessarily optimal. In contrast, our initial solution (shortest path) is optimal, but it is not feasible. It is infeasible because some of the capacity constraints are violated. In the dual simplex method, the LP problem starts (better than) optimal and infeasible. Successive iterations are designed to move toward feasibility without violating optimality. At the iteration when feasibility is restored, the algorithm ends.

We are proposing the use of revised dual simplex method, as we shall take advantage of the way that the basis matrix is being calculated. In the dual simplex method, the starting tableau must have an optimum objective row with at least one infeasible (< 0) basic variable. To maintain optimality and, simultaneously, to move toward feasibility at each new iteration, the following two conditions are required:

Dual feasibility condition. The leaving variable, x_r, is the basic variable having the largest negative value, with ties broken arbitrarily. If all the basic variables are non-negative, the algorithm ends.

Dual optimality condition. The entering variable is determined from among the nonbasic variables as the one corresponding to:

$$\min_{\text{Nonbasic } x_j} \left\{ \left| \frac{z_j - c_j}{\alpha_{rj}} \right|, \alpha_{rj} < 0 \right\}$$

where α_{rj} is the constraint coefficient of the tableau associated with the row of the leaving variable x_r and the column of the entering variable x_j. Ties are broken arbitrarily.

Product Form of The Inverse of a Matrix

In the simplex method, the successive bases, B and B_{next}, differ only in one column vector, resulting from interchanging the entering and the leaving vectors. This situation is ideal for the application of the product form of inversion.

Given B^{-1}, we can compute $\mathrm{B}_{\text{next}}^{-1}$ using the formula

$$\mathbf{B}_{\text{next}}^{-1} = \mathbf{E}\mathbf{B}_{\text{next}}$$

To compute matrix $\mathbf{E}$, let $\mathbf{P}_j$ and $\mathbf{P}_r$ be the entering and the leaving vectors in the current simplex iteration. Then $\mathbf{E}$ is defined as an m-identity matrix whose r^{th} column is replaced by

$$\zeta = \frac{1}{(\mathbf{B}^{-1}\mathbf{P}_j)_r} \begin{bmatrix} -(\mathbf{B}^{-1}\mathbf{P}_j)_1 \\ -(\mathbf{B}^{-1}\mathbf{P}_j)_2 \\ \vdots \\ +1 \\ \vdots \\ -(\mathbf{B}^{-1}\mathbf{P}_j)_m \end{bmatrix} \leftarrow r^{\text{th}} \text{ place}$$

provided that $(\mathbf{B}^{-1}\mathbf{P}_j)_r \neq 0$. If $(\mathbf{B}^{-1}\mathbf{P}_j)_r = 0$, then $\mathbf{B}^{-1}$ does not exist.

Inverse Matrix

Given A a nonsingular matrix of size n, we would like to compute its inverse, such that

$$\mathbf{A}\mathbf{A}^{-1} = I$$

We can partition the matrix A of size n as follows, given that A_{11} is non-

singular.

$$\mathbf{A} = \left[\begin{array}{c|c} \mathbf{A}_{11} & \mathbf{A}_{12} \\ (p \times p) & (p \times q) \\ \hline \mathbf{A}_{21} & \mathbf{A}_{22} \\ (q \times p) & (q \times q) \end{array}\right]$$

Let $\mathbf{B} = \mathbf{A}^{-1}$, we can partition $\mathbf{B}$ such that

$$\mathbf{B} = \left[\begin{array}{c|c} \mathbf{B}_{11} & \mathbf{B}_{12} \\ (p \times p) & (p \times q) \\ \hline \mathbf{B}_{21} & \mathbf{B}_{22} \\ (q \times p) & (q \times q) \end{array}\right]$$

using the result from [56], the value of $\mathbf{B}_{11}$, $\mathbf{B}_{12}$, $\mathbf{B}_{21}$ and $\mathbf{B}_{22}$ are given by:

$$\mathbf{B}_{11} = \mathbf{A}_{11}^{-1} + (\mathbf{A}_{11}^{-1}\mathbf{A}_{12})\mathbf{D}^{-1}(\mathbf{A}_{21}\mathbf{A}_{11}^{-1})$$

$$\mathbf{B}_{12} = -(\mathbf{A}_{11}^{-1}\mathbf{A}_{12})\mathbf{D}^{-1}$$

$$\mathbf{B}_{21} = -\mathbf{D}^{-1}(\mathbf{A}_{21}\mathbf{A}_{11}^{-1})$$

$$\mathbf{B}_{22} = \mathbf{D}^{-1}$$

where

$$\mathbf{D} = \mathbf{A}_{22} - \mathbf{A}_{21}(\mathbf{A}_{11}^{-1}\mathbf{A}_{12})$$

In our problem, we found that the matrix that we want to invert has the following structure:

$$\mathbf{A}_{11} = \mathbf{I}$$

$$\mathbf{A}_{21} = \mathbf{0}$$

$$\mathbf{A}_{22} = \mathbf{I}$$

$\mathbf{A}_{12}$ is dependent on the network topology.

For this special case, we substituted the partitioned matrix As and found that matrix B, which is the inverse of A, as follows:

$$\mathbf{B} = \mathbf{A}^{-1} = \left[\begin{array}{c|c} \mathbf{I} & -\mathbf{A}_{12} \\ (p \times p) & (p \times q) \\ \hline \mathbf{0} & \mathbf{I} \\ (q \times p) & (q \times q) \end{array}\right]$$